НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

До захисту допущено:

Завідувач кафедри

_____ Олександр Коваль

«___»_____2020 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Інформаційні технології моніторингу довкілля»

спеціальності 122 «Комп'ютерні науки та інформаційні технології»
на тему: «Створення програмних засобів формування бази даних інженерних мереж на базі ГІС технологій»

Виконав (-ла):
студент (-ка) IV курсу, групи ТМ-61
Козачук Олександр Володимирович _____

Керівник:
Асистент,
Швайко Валерій Григорович _____

Рецензент:
Кандидат технічних наук,
Шевченко Олена Миколаївна _____

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2020 року

АНОТАЦІЯ

Метою цієї роботи є розробка зручної програмної системи для формування та перегляду геоінформаційної бази даних.

Для оцінки можливостей додатку були перенесені в геоінформаційну БД інженерні мережі Національного Технічного Університету України «Київський Політехнічний Інститут імені ІГОРЯ СІКОРСЬКОГО».

ГБД містить інформацію про мережі енерго-, водо- та газопостачання. Також в базі даних зберігається інформація про лічильники та їх показники. БД розміщена у вигляді сервісу об’єктів, тому може бути розширена та доповнена іншими модулями.

Загальний обсяг роботи: 48 сторінок, 27 ілюстрацій та 15 бібліографічних найменувань.

Ключові слова: Інженерні мережі, Геоінформаційна база даних, програмна система, лічильники, додаток, сервіс об’єктів.

ABSTRACT

The purpose of this work is to develop a convenient software system for creating and viewing a geoinformation database.

To assess the capabilities of the application, engineering networks were transferred to the geoinformation database of the National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"

GBD contains information about energy, water and gas engineering networks. The database also stores information about counters and their indicators. The database is placed as an feature service, so it can be expanded and supplemented with other modules.

Total volume of work: 48 pages, 27 illustrations and 15 references.

Key words: engineering networks, geodatabase, software system, counters, application, feature service

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти перший рівень

Напрям підготовки 122 Комп’ютерні науки та інформаційні технології

Спеціалізація Інформаційні технології моніторингу довкілля

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр Коваль
(підпис)
”___”_____2020р.

ЗАВДАННЯ
на дипломну роботу студенту

_____Козачуку Олександру Володимировичу_____
(прізвище, ім’я, по батькові)

1. Тема роботи ____«Створення програмних засобів формування бази даних інженерних мереж на базі ГІС технологій»_____
керівник роботи _____Швайко Валерій Григорович, асистент_____
(прізвище, ім’я, по батькові науковий ступінь, вчене звання)
затверджена наказом вищого навчального закладу від ”25” травня 2020р. № **1168-с**
2. Строк подання студентом роботи _____

3. Вихідні дані до роботи ____Мобільний додаток для платформи iOS, а____ також Геоінформаційна база даних у форматі *.gdb._____
Засоби розробки Swift, UIKit, ArcGIS Runtime SDK, Середовища розробки Xcode, ArcGIS Desktop_____
4.Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Проаналізувати можливості ГІС технологій та побудову інженерних мереж. Спроектвати та розробити мобільний додаток що дав би змогу редагувати параметри інженерної мережі та наповнити зв’язні таблиці просторових об’єктів_____
5. Перелік ілюстративного матеріалу «Ієрархія інженерної мережі», «Приклад бізнес таблиці класу полігональних об’єктів», «Класи просторових об’єктів», «Приклад просторового об’єкту - Мультиточки», «Приклад просторового об’єкту – Мультипатчу», «Робочий процес що використовується для

створення сервісу об'єктів», «Інтерфейс додатку», «Діаграма прецидентів», «Код класів системи», «Структура архітектури MVC»

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання ”___”___грудня___2019 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної з/п роботи	Термін виконання етапів роботи	Примітки
1.	Затвердження теми роботи	03.02.20	
2.	Вивчення та аналіз задачі	13-19.04.20	
3.	Розробка архітектури та загальної структури системи	20-26.04.20	
4.	Розробка структур окремих підсистем	27.04-03.05.20	
5.	Програмна реалізація системи	04-13.05.20	
6.	Оформлення пояснювальної записки	14-16.05.20	
7.	Захист програмного продукту	17.05.20	
8.	Передзахист	09.06.20	
9.	Захист	17.06.20	

Студент _____ Козачук О. В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Швайко В. Г.
(підпис) (прізвище та ініціали)

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

ГІС — геоінформаційна система.

ГБД — геоінформаційна база даних.

БД — база даних.

СКБД — система керування базою даних

ВСТУП

Зі стрімким розвитком рівня індустрії та промисловості розвиваються міста та мегаполіси. Більша частина людей обирають міста задля комфортного життя, свого та майбутніх поколінь. А також з їх розвитком розвиваються та ускладняються інженерні мережі. Інженерні мережі — це сукупність систем будівлі, що допомагають забезпечувати його життєдіяльність і функціонування відповідно до призначення. Більшість з них зберігаються на паперових кресленнях, які важко масштабуються, змінюються та розповсюджуються. Це викликає певні труднощі для осіб, які мають справу з інженерними мережами. З розвитком інформаційних технологій з'явилась можливість зберігати, обробляти, та аналізувати мережі використовуючи обчислювальну техніку, таку як комп'ютери та мобільні пристрої. Це набагато простіше, адже користуватися можливостями, які відкрились може кожен, незалежно від освіти, місцеперебування та віку.

Енергетичний менеджмент – це певний вид діяльності, що дозволяє значно оптимізувати обсяги енерговитрат. Це нова галузь знань і досвіду людини, бурхливе формування якої спостерігається сьогодні. Адже ці знання сьогодні, як ніколи, потрібні задля майбутнього наших поколінь. Фахівець, який добре розбирається в енергоменеджменті, може розробляти ефективні системи та засоби контролю за енергоспоживанням, та захищати наше довкілля від надмірного забруднення, створити прозорість у сфері використання енергоресурсів. Народившись у розвинутих країнах США, Західної Європи та Японії в 70-х роках XX ст. як шлях до подолання енергетичної кризи та побудови енергоефективної економіки, ця нова самостійна система знань дуже активно розвивається практично в усіх країнах світу. Вона є синтезом гуманітарних та технічних знань і досвіду, тобто енергетичний менеджмент формується на перехресті менеджменту та технологій [15].

Основною метою енергоменеджменту є забезпечення високої енергоефективності господарювання при оптимальному використанні людських та

матеріальних ресурсів та мінімізація впливу використання енергоресурсів на навколишнє середовище. У результаті негативного впливу невпинно зростаючого енергоспоживання в багатьох районах світу вже сьогодні створилася дуже небезпечна екологічна ситуація. Розв'язання цієї проблеми та займаються фахівці, які вивчають енергетичний менеджмент роками, та мають певний досвід роботи з усуненням негативних наслідків надмірного використання енергоресурів. Для досягнення цієї цілі необхідне впровадження системи законодавчих, правових, технічних, економічних, наукових та інформаційних заходів, які спрямовані на оптимальне використання енергетичних ресурсів.

1 Постановка задачі створення програмних засобів формування бази даних інженерних мереж на базі ГІС технологій

Метою дослідження є аналіз перспектив та варіантів використання Геоінформаційних та мобільних технологій в енергетиці для побудови інженерних мереж.

ArcGIS - сімейство геоінформаційних програмних продуктів американської компанії ESRI. Застосовуються для земельних кадастрів, в задачах землеустрою, обліку об'єктів нерухомості, систем інженерних комунікацій, геодезії та надрокористування та інших областях.

Інтегровані інженерні мережі в геоінформаційну систему швидше переглядати, аналізувати та оптимізувати їх, розширює можливість для їх обліку. Вже зараз інженерні мережі що зберігаються на серверах спрощують аналіз використання енергоресурсів та дають інженеру простий інструмент для оптимізації споживання певними ресурсами.

Було поставлено задачу створити програмний продукт що дав би змогу інженеру швидко переглядати, змінювати та аналізувати інженерну мережу та її показники у реальному часі.

1.1.Мета створення програми формування бази даних інженерних мереж

Метою даної дипломної роботи є створення програмного продукту що дав би змогу інженеру швидко переглядати, змінювати та аналізувати інженерну мережу та її показники у реальному часі.

Ця задача складається з наступних етапів:

- Створення структури ГБД
- Наповнення БД відповідними Інженерними мережами
- Створення зв’язків між мережами та лічильниками
- Розробка мобільного додатку маніпуляції з мережами

Програмна система реалізує функціонал для зручної маніпуляції з ГБД та інженерними мережами. Програмний продукт реалізує наступний функціонал:

- Ідентифікація лічильника або люка що обрав користувач
- Відображення детальної інформації по обраному елементу
- Відображення зв’язних таблиць елемента, а у випадку лічильника це показники
- Редагування та видалення обраних елементів
- Додання нового лічильника та нових показників

1.2 Вхідні дані до системи

Розроблена геоінформаційна база даних має зберігати інформацію про розташування інженерних мереж, люків, лічильників з прив’язкою до географічних просторових координат, а також таблиці з додатковою інформацією та показниками лічильників. Вхідними даними до цієї геоінформаційної бази даних є положення лічильників, інженерних мереж та додаткова інформація, що зображено на технічних картах та атрибутиці лічильників.

Відповідні вхідні дані до системи попередньо завантажуються як сервіс об’єктів на власний сервер або ж на ArcGIS Online. Після завантаження додатку на мобільний пристрій, система одразу готова до використання.

Вхідними даними для мобільного додатку є координати точки в момент натиску на екран смартфону, а також жести масштабу та повороту.

Завдяки тому що база даних зберігається на сервері, мобільний додаток не займає багато місця на смартфоні та користувач має повний доступ до системи при наявності підключення до мережі інтернет.

1.3 Компоненти системи

Проаналізувавши завдання дипломної роботи були виділені основні компоненти для робочої системи:

- Мобільний додаток
- Геоінформаційна база даних
- Хмарний сервіс для розміщення сервісу об’єктів

База даних може зберігатись та редагуватись як локально на комп’ютері, так і на сервері. Також, перед публікацією ГБД обов’язково мають бути сформовані всі зв’язки між таблицями. Мобільний додаток забезпечує зручний доступ для відображення та редагування Геоінформаційної бази даних.

1.4 Потенційні користувачі

Система, що реалізована при виконанні дипломної роботи може використовуватись людьми яким необхідний програмний продукт для перегляду, аналізу та редагування інженерних мереж. Для персонального налаштування даної системи необхідно розробити структуру геоінформаційної бази даних та завантажити її на сервер. Варіант, що був розроблений в даній роботі може бути використаний енергоменеджерами НТУУ «КПІ ім. Ігоря Сікорського», або ж, при наступному розширені та доповнені, користувачами енерго-, водо- та газопостачання. Також система може бути корисна людям яким необхідно переглядати та аналізувати їх ГБД з мобільного пристрою.

2 Аналіз доцільності використання ГІС технологій для побудування інженерних мереж

2.1 Аналіз схожих систем

При проектуванні системи було знайдено та проаналізовано декілька схожих систем, та виділені певні переваги та недоліки:

DataCollection for IOS – система, що знаходиться у відкритому доступі на порталі ArcGISonline. Виконує функції примітивного збирача даних.

Переваги та недоліки DataCollectionfor IOS

Таблиця 2.1

Переваги	Недоліки
• Готовий одразу до використання • Знаходиться у відкритому доступі • Легкий у освоєнні • Має обширну документацію	• Може працювати лише з веб мапами • Потребує спеціального програмного забезпечення для інсталяції • Для великих об’ємів даних необхідна платна підписка • Працює лише під певну операційну систему

.NET DataCollection –opensource система для збереження та відображення просторових даних.

Переваги та недоліки .NET DataCollector

Таблиця 2.2

Переваги	Недоліки
• Легко масштабується • Працює з різними типами просторових даних • Кросплатформний	• Потребує постійного інтернет з’єднання • Важчий у використанні, у порівнянні з попереднім

• Дані зберігаються на сервері	

ArcGISDesktop – програмний продукт від компанії Ersi для створення і редагування геоінформаційної бази даних.

Переваги та недоліки ArcGISDesktop

Таблиця 2.3

Переваги	Недоліки
• Має великий інструментарій • Є професійним інструментом для редагування просторових даних • Набагато швидший у порівнянні з попередніми	• Дані зберігаються оффлайн • Потребує підписки або купівлі дорогого програмного пакету • Працює лише на windows системах • Потребує потужного апаратного забеспечення

2.2 Інженерні мережі

Інженерні мережі - це сукупність елементів, які роблять роботу і життя в будівлях найбільш комфортною. Без таких компонентів неможливо назвати споруду повноцінним, і придатним для експлуатації. Це комплекс систем будівлі, що забезпечують його життєдіяльність і функціонування відповідно до його призначення. Сучасна будівля складно уявити без інженерних систем: якщо для 1900 року опалення дровами і колодязь для водопостачання були у більшості домоволодінь, то в даний момент все з точністю навпаки - будівель з пічним опаленням і без каналізації з водопостачанням знайти все важче

Інженерні системи бувають як внутрішніми, так і зовнішніми - з назви зрозуміло, що інженерні системи, що знаходяться всередині будівлі відносяться до внутрішніх, а що знаходяться за межами зовнішньої стіни відносяться до зовнішніх інженерних систем.

2.2.1 Опис інженерних мереж

Геоінформаційні системи в сфері інженерних мереж відіграють велику роль. Функції, які вони виконують це функція проектування, інвентаризації, моделювання. Також, важливою функцією можна вважати функцію інформаційної підтримки експертних оцінок і прийняття рішень. Геоінформаційні системи експлуатують переважно для використання інженерних мереж. Вони є досить непоганими інформаційно-довідковими системами. ГІС такого класу має ряд основних особливостей :

- в наявної моделі мережі є імітація стану елементів і ділянок мережі;
- описаний життєвий цикл мережі та її елементів;
- існування геометричного уявлення мережі на карті або плані з розмірними прив’язками. І це все придатне для креслярського уявлення і завдань узгодження;
- детальний опис технічних параметрів елементів мережі;
- існування певних коштів для документообігу.

Інженерна мережа – це основний компонент, з яким працюють користувачі під час адміністрування та управління інженерними та телекомунікаційними мережами в системі ArcGIS. Водночас, поруч із заснованою на сервісах трансакційною моделлю, правилами атрибутів, інструментами, за допомогою яких можна редагувати і втілювати в життя всі свої побажання, вона дозволяє користувачам повністю аналізувати і моделювати комплексні мережеві системи газопостачання, електропостачання, водопостачання, телекомунікаційні мережі, систему стічних вод, каналізації тощо.

Аналіз та управління мережевих даних відбувається в модулі модулі ArcGIS Utility Network Management Extension ArcGIS Enterprise. Цей модуль дозволяє працювати з всіма пристроями або додатками з підтримкою веб-сервісу по заснованої на сервісах архітектурі.

Інженерна мережа – це функціонал ArcGIS, який призначений для створення моделей таких інженерних мереж, як газ, електрика, каналізація, водопостачання та телекомунікації. Вона дозволяє проводити моделювання всіх компонентів системи, таких як труби, дроти, зони, клапани, і дає можливість моделювати реальну поведінку об’єктів мережі.

Використання інженерної мережі допомагає робити наступне :

- вивчати по якому принципу з’єднані об’єкти в мережі;
- створювати і проводити редагування об’єктів, що моделюють всі типи мережевого обладнання;
- створювати робочі види, які наглядно показують як динамічні пристрої мережі налаштовані на даний момент часу;
- виконувати трасування проходження по мережі джерел, таких, як електричний струм, вода або газ;
- проводити аналіз мережі в екстремальних умовах, наприклад, штормами або виходом з ладу обладнання.

2.2.2 Структура Інженерних мереж

За допомогою інженерної мережі можна змоделювати те, як з’єднані всі компоненти інженерної системи, користувачі мають змогу розумно обробляти щільні добірки інженерних просторових об’єктів і аналізувати трасування мережі з урахуванням ієрархії (Рисунок 2.1).

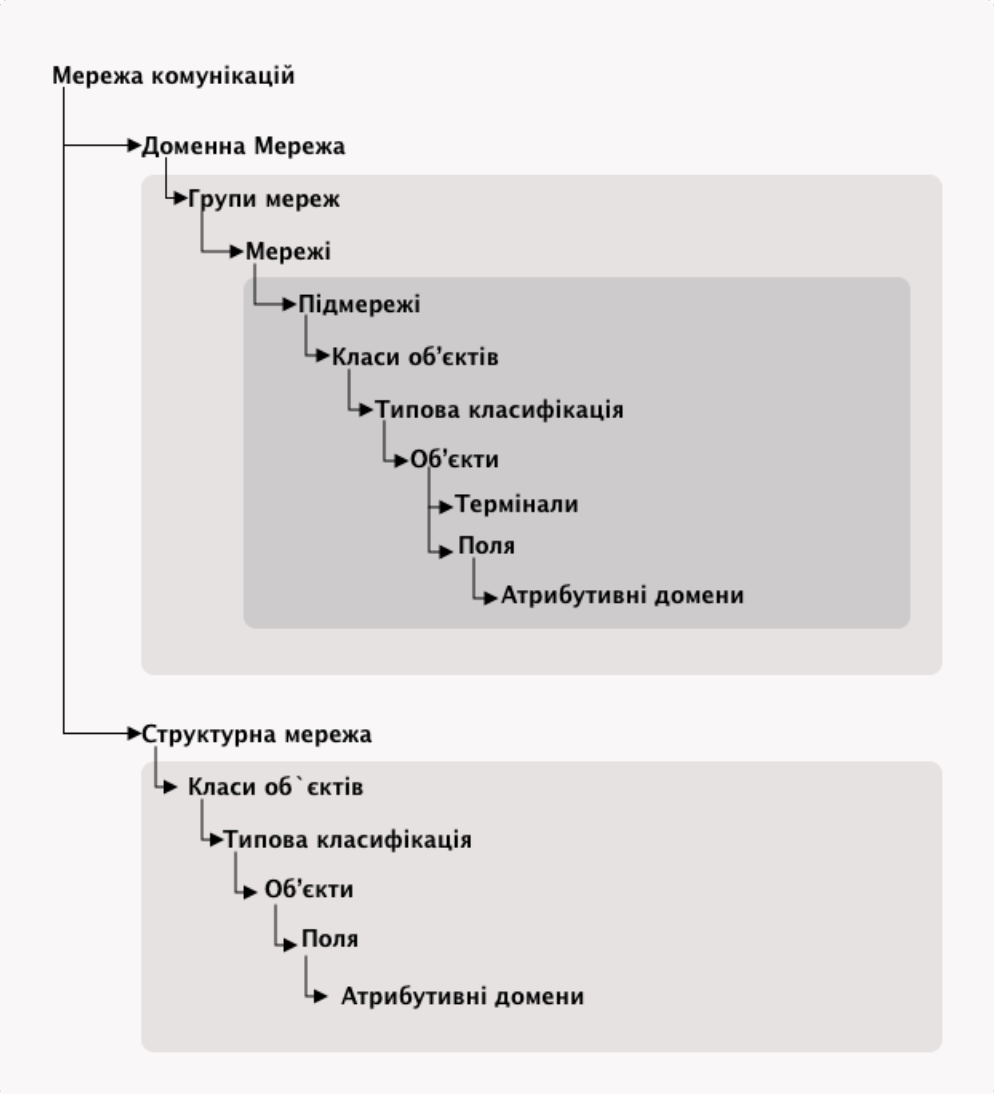

Рисунок 2.1. Ієрархія інженерної мережі

Під час створення інженерної мережі генерується попередньо налагоджений набір класів просторових об’єктів. З самого початку створюються три класи просторових об’єктів, вони допомагають в роботі різних типів інженерних просторових об’єктів. Сукупність цих всіх класів називають структурною мережею.

В інженерній мережі містяться одна або декілька спеціалізованих мереж і одна структурна мережа. Вони об’єднують стандартизовані класи, які були створені під час операції зі створення та додавання спеціалізованої мережі, яка налаштована на моделювання всіх систем, які знаходяться в інженерній мережі [10].

Спеціалізована інженерна мережа на початку об’єднує всі об’єкти в групи рівнів та рівні. Рівні являє собою ієрархію того, як мережа доставляє ресурси, наприклад, електрику, природний газ або воду. Рівень, в основному, це тиск або

напруга. На прикладі електричної мережі – вона може бути високовольтна, з середньою напругою та споживча. Певні види аналізу мають відбуватись лише на одному ієрархічному рівні. Окрім цього, рівень може позначати частину мережі, яка може бути відділена від іншої, наприклад, зони, які розділені запірним клапаном в системах з тиском. За допомогою визначених рівнів можна встановлювати обмеження для допустимих об’єктів для кожного рівня. Також можна задавати діапазон аналізу трасування мереж.

Потік ресурсів в інженерній мережі контролюється спеціальними пристроями. Найпоширенішими з них є клапани і вентилі. Підмережою можна називати розширення доставлення ресурсів в заданий момент часу. Підмережа відповідає зоні тиску для газових і водопровідних мереж або мереж для електропостачання.

В спеціалізованій мережі в п’ять класів просторових об’єктів згруповані всі просторові об’єкти. Ці класи містять просторові об’єкти для пристроїв, ліній підмереж, збірок пристроїв, ліній і з’єднань, де з’єднуються частини мережі. В кожній спеціалізованій мережі є ці класі об’єктів і вони оптимізовані для високопродуктивного відображення, аналізування та редагування.

При додаванні просторових об’єктів у вас з’являються розширені можливості визначення того, як саме пов’язані між собою інженерні об’єкти. При розриві просторового об’єкта або лінії в одному і тому самому ж місці вони можуть бути автоматично з’єднані. Окрім цього ви можете самі задати логічний взаємозв’язок між пристроями, які розташовані окремо один від одного. Такий тип зв’язності особливо корисний в регіонах з високою щільністю об’єктів інженерних карт.

Безліч інженерних приладів знаходяться в сховищах, будках або дворах. В інженерних мережах є можливість згрупувати просторові об’єкти в контейнери, так щоб на карті був відображений комплексний об’єкт, а не безліч дрібних об’єктів. Доступ до просторових об’єктів можна отримати за запитом або у вигляді діаграм.

В інженерних мережах існує об’ємна система класифікації типів, що допомагає отримати доступ до кожного типу інженерного просторового об’єкту. Ця операція

виконується через поля ASSETGROUP та ASSETTYPE щоб надати об’єктам класифікацію.

Визначення умов зв'язності мережі, визначення символів просторовим об'єктам, трасування, налаштовуються безпосередньо в класах об'єктів мережі, та використовує класифікаційну систему. Це надає функціонал інженерній мережі ідентифікувати низькорівневу класифікацію задіючи невелику кількість класів просторових об'єктів.

В інженерних мережах міститься інформація про пристрої, що підключені до певних структур, як наприклад сховища або стовпи. В інженерних мережах є можливість визначення приналежності до структури, та таким чином показувати відфільтровані списки для територій та підмереж

Користувач вказує правила для об'єктів певних мереж, що можуть бути об’єднані структурними об’єктами. Ці зв’язки можуть бути переглянуті за запитом або на карті.

2.2.2 ГІС в сфері експлуатації інженерних мереж

Перевагами, які відрізняють усі автоматизовані системи є багатокористувацька робота, централізоване зберігання інформації, подання даних підприємства в електронному виді, складання звітів. Окрім цих переваг, ГІС в інженерних мережах також дає:

- інженерна мережа подається у вигляді моделі, це дозволяє проводити її аналіз методами теорії графів. Після того, як проаналізована топологія мережі та проведені всі необхідні топологічні розрахунки, відкривається можливість провести вже технологічні розрахунки. Технологічними розрахунками можна вважати розрахунок рівня тиску в трубопроводі або сили струму короткого замикання. Ці розрахунки, в основному, і є основною можливістю ГІС в інженерних мережах, також саме це і відрізняє їх від ГІС інших призначень.

ГІС тут – швидше лише одна із важливих складових системи по експлуатації на підприємстві.

- Прив’язка до реальної географії – це коли відображається точна топологія мережі на плані місцевості або міста. Але не завжди обов’язково і доцільно використовувати точність в завданні геодезичних координат, тому що для певних випадків вона не потрібна, а сам процес детального промальовування мережі сильно уповільнить етап розрахунків і швидкість нашої роботи. До прикладу, вигини та повороти провідників в електричній мережі не мають впливу на силу струму, який протікає в них. В теплових мережах все працює навпаки, наявність вигинів створює гідравлічний опір мережі, але це можна врахувати як просте завдання параметру.
- Модель реальної мережі можна узагальнити. До прикладу, можна відобразити декілька проводів(трифазної електричної мережі) , які йдуть паралельно один до одного однією лінією та з певним заданим чином атрибутів. В кінці , на результат це не вплине, зате дозволить підвищити швидкість введення даних. Таким самим чином можна узагальнювати певні ділянки мережі та здійснювати розрахунки для них, як для одного елементу.
- Робота в ГІС в інженерних мережах має можливість істотно полегшити задачу введення параметрів, адже вибір необхідних для роботи об’єктів відбувається графічним методом, а не лише з таблиць БД. Для роботи це дуже зручно, виділяти потрібні ділянки в мережі і ставити для всіх відразу однакові значення параметрів (якщо в цьому є необхідність), особливо якщо таких ділянок багато.
- Графічна вказівка на помилки, які були отримані в результаті розрахунків або під час введення атрибутивної інформації дуже полегшує знаходження проблемного місця в мережі. До прикладу, можна підсвітити кольором, відмінним від кольору тексту, таку ділянку. Також однакове графічне відображення вихідних даних і результатів, які вийшли в процесі розрахунків, підвищує наочність моделі.

2.3 Геоінформаційна система

База геоданих ArcGIS являє собою набір географічних наборів різних типів, всі вони зберігаються в місці файлової системи, яке має загальнодоступний характер – база даних Microsoft Access або багатокористувацька реляційна база даних (наприклад, Oracle , PostgreSQL, Microsoft SQL Server, IBM DB2 або Informix). Їхньою перевагою є те, що вони можуть масштабуватись від маленьких баз даних, що зосереджені лише на файлах, з одним користувачем, до великих за масштабністю галузевих, групових і корпоративних баз геоданих, які мають режим доступу, який розрахований на багато користувачів [7].

База геоданих являє собою більше, аніж просто колекцію наборів даних. Навіть термін «база геоданих» в ArcGIS може мати декілька значень:

- База геоданих – вона є «рідною» структурою даних для ArcGIS; це основний формат даних, який використовують для зручного редагування і управління даними. Не дивлячись на те, що ArcGIS працює з інформацією географічного характеру, що може знаходитись в різних форматах географічних систем (ГІС), основні та найпотужніші його функціональні можливості використовуються в базах геоданих.
- База геоданих – являє собою фізичне сховище для географічної інформації – в основному використовує файлову систему або СКБД. Користувач може отримати доступ і мати можливість працювати з фізичним екземпляром наборів даних безпосередньо в ArcGIS, або ж в системі управління базами даних за допомогою SQL.
- База геоданих має свою модель транзакцій, що дозволяє управляти робочими потоками ГІС-даних.

- Бази геоданих реалізуються інформаційною моделлю для управління та відображення геоінформації. Ця інформаційна модель є всебічною та реалізується серією простих таблиць з введеними даними, що містять атрибути, набори растрів та класи просторових об’єктів. Окрім цього, розширені об’єкти ГІС-даних можуть додавати ГІС-поведінку, інструменти для роботи з численними просторовими відносинами основних просторових об'єктів, растрів і атрибутів
- Програмна логіка бази геоданих забезпечує загальну логіку додатка, використовувану у всій ArcGIS для доступу і роботи з усіма географічними даними в різних файлах і форматах. Що, безсумнівно, включає підтримку роботи з самою базою геоданих. А також роботу з шейп-файлами, файлами САПР, грід, TIN, даними САПР, зображеннями і багатьма іншими джерелами ГІС-даних.

2.3.1 Архітектура ГБД

Модель зберігання даних в базі геоданих заснована на концепції реляційних баз даних і реалізує весь функціонал системи управління базою даних. Прості таблиці і добре певні типи використовуються для зберігання схеми, правил, базових і просторово-атрибутивних даних для кожного набору географічних даних. Це дозволяє використовувати формалізовану модель для зберігання ваших даних і роботи з ними. Завдяки такому підходу, мова структурованих запитів (SQL) - набір реляційних функцій і операторів - може бути використаний для створення, зміни та виконання запитів до таблиць і їх елементів даних[14].

Parcels

FID	Shape *	PROPERTY_J *	Res	Zoning_simple	SHAPE_Length	SHAPE_Area
1	Полигон	5001	Non-Residential	<Null>	3597.780813	112552.418591
2	Полигон	5002	Non-Residential	<Null>	814.855837	18488.417709
3	Полигон	1003	Residential	Residential	489.655523	12815.591379
4	Полигон	1004	Residential	Residential	521.761248	14036.135346
5	Полигон	1005	Residential	Residential	453.479649	9816.352665

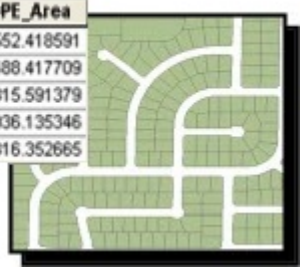

Рисунок 2.2 Приклад бізнес таблиці класу полігональних об’єктів

Бази геоданих реалізуються за допомогою багаторівневої архітектури додатків, яка використовується в розвинених СКБД -додатках; нічого специфічного не можна сказати про цю реалізації. Багаторівнева архітектура бази геоданих іноді називається об'єктно-реляційної моделлю. Об'єкти бази геоданих існують у вигляді рядків в таблицях СКБД, які мають ідентифікацію та поведінку, які підтримуються логікою додатка для бази геоданих. Оскільки існує поділ логіки програми та логіки зберігання інформації, це забезпечує підтримку різних форматів даних та СКБД.

Ядром бази геоданих є стандартна реляційна схема бази даних (набір стандартних таблиць бази даних, типів полів, індексів та інших об'єктів бази даних). Схема існує у вигляді набору системних таблиць бази геоданих в СКБД, які визначають цілісність і поведінку географічної інформації. Ці таблиці зберігаються або у вигляді файлів на диску або в вмісті СКБД, такий як IBM Informix, IBM DB2, Oracle, Microsoft SQL Server або PostgreSQL

Добре певні типи полів використовуються для зберігання традиційних атрибутів в таблицях. Коли база геоданих зберігається в СКБД, просторові уявлення, такі як векторні і растрові дані, які зазвичай зберігаються за допомогою розширених просторових типів.

У базі геоданих існує два основних набору таблиць - це системні таблиці та таблиці наборів даних.

- Таблиці наборів даних - кожен набір даних в базі геоданих зберігається в одній або декількох таблицях. Таблиці наборів даних для управління даними працюють з системними таблицями.
- Системні таблиці - відстежують вміст кожної бази геоданих. Вони описують структуру бази геоданих, в якій описані всі визначення, правила і відносини наборів даних. Ці системні таблиці містять і керують усіма метаданими, необхідними для реалізації властивостей бази геоданих, правил перевірки даних і поведінки.

2.3.2 Класи просторових об'єктів

Класи просторових об'єктів - однорідні сукупності однотипних об'єктів, кожен з яких має однакову просторове уявлення, у вигляді точок, ліній, або полігонів, і загального набору атрибутивних полів, наприклад, лінійний клас просторових даних для подання осьових ліній доріг. Чотири основних типи класів просторових даних - це точки, лінії, полігони і анотації (рис 2.3).

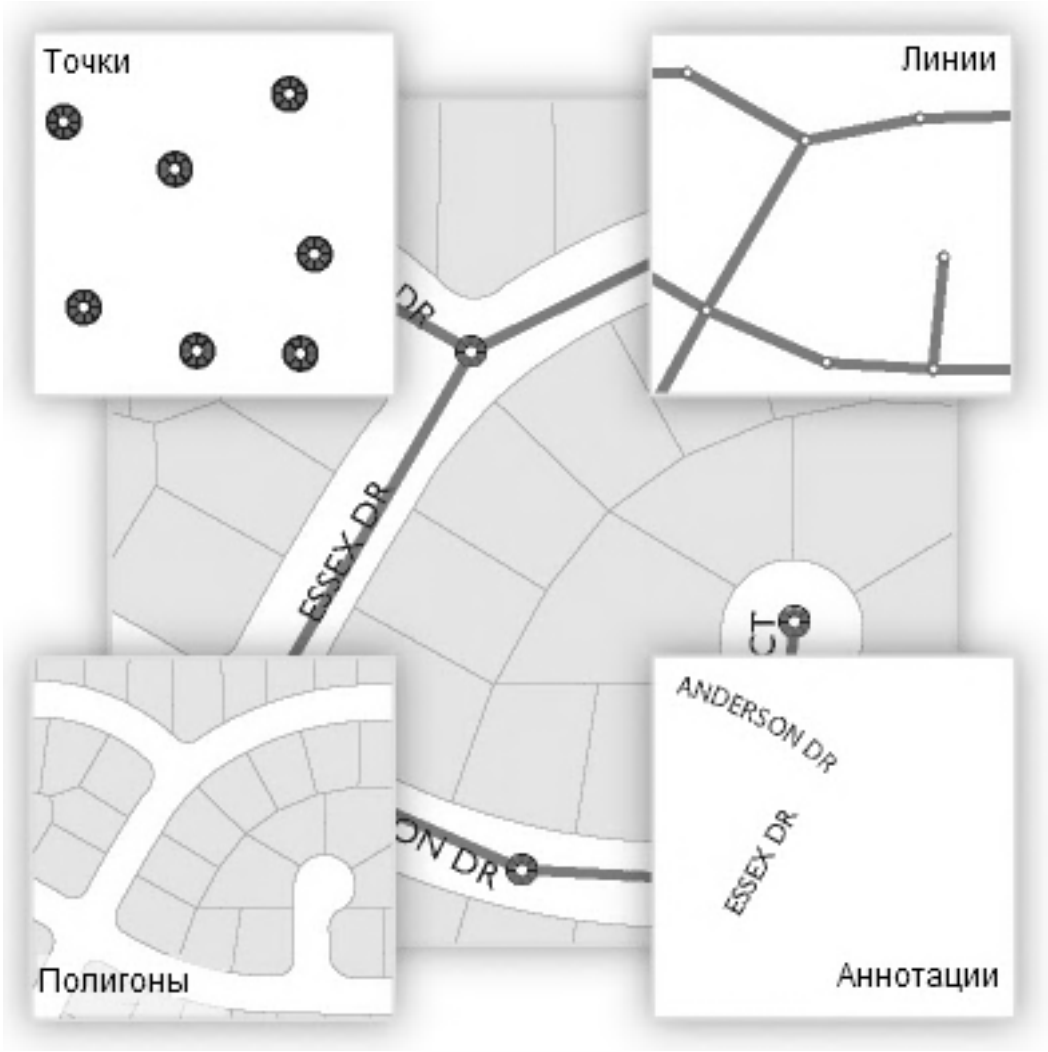

Рисунок 2.3 Класи просторових об’єктів

Векторні об'єкти (географічні об'єкти з векторної геометрією) різнобічні і є часто використовуваними географічними типами даних, добре підходять для представлення об'єктів з дискретними межами, наприклад, адресу, адміністративні кордони і земельні ділянки. Просторовий об'єкт - це об'єкт, який зберігає своє

географічне уявлення, представлене зазвичай у вигляді точки, лінії або полігону, в якості одного з властивостей (полів) в рядку. ArcGIS класи просторових даних - однотипні сукупності об'єктів із загальним просторовим поданням і набором атрибутів, що зберігаються в таблиці бази даних, наприклад, клас лінійних об'єктів, що представляє осьові лінії доріг.

Як правило, класи просторових об'єктів є тематичними наборами точок, ліній або полігонів, але в дійсності існує сім типів класів просторових об'єктів. Перші три підтримуються в базах даних і базах геоданих. Решта чотири підтримуються тільки в базах геоданих.

- Точки: просторові об'єкти, які дуже малі, щоб позначати їх лініями або полігонами, а також точкові розташування (точки GPS).
- Лінії: відображають форму і місце розташування географічних об'єктів, що занадто вузькі для відображення у вигляді полігонів. Лінії також використовуються для відображення об'єктів, що мають довжину і не мають площі, таких як ізолінії та кордони.
- Полігони: набір багатосторонніх майданних об'єктів, що становлять форму і місце розташування однорідних типів просторових об'єктів, таких як адміністративні райони, округи, ділянки землі, типи грунту та зони землекористування.
- Анотації: підпис на карті, яка містить параметри відображення тексту. Наприклад, крім текстового рядка кожної анотації, там зберігаються і інші властивості - наприклад точки фігури для розміщення тексту, його шрифт і точковий розмір, а також інші властивості відображення. Анотація може також бути пов'язаної з надписується об'єктами і може містити підкласи.
- Об'єкти-розміри: спеціальний тип анотації, що показує специфічні довжини або відстані, наприклад, для вказівки довжини сторони будівлі, ділянки землі або відстані між двома об'єктами. Розміри найчастіше використовуються для дизайнерських і інженерних задач в ГІС.

- Мультиточки: просторові об'єкти, що складаються з більш ніж однієї точки. Мультиточка часто використовуються для управління масивами дуже великих сукупностей точок, таких як, наприклад, кластери точок LiDAR, які можуть містити мільярди пунктів. Використання одного запису для такої точкової геометрії неприпустимо. Об'єднання таких даних в групи записів об'єктів-мультиточок надає можливість базі геоданих управляти масивними наборами точок (рис. 2.4)

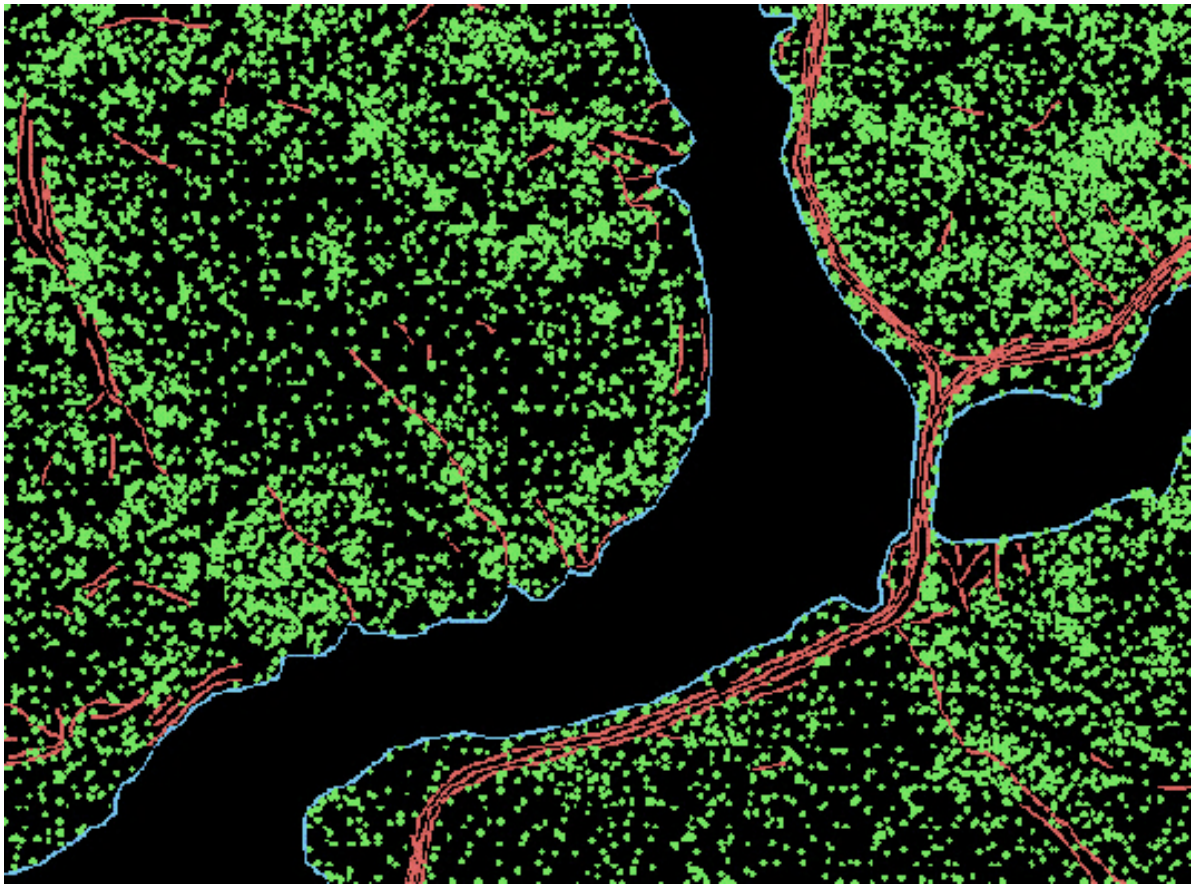

Рисунок 2.4 Приклад просторового об’єкту - Мультиточки

- Мультипатчі: 3D-геометрія, використовувана для подання зовнішньої поверхні, або оболонки, об'єктів, які займають дискретну область або обсяг в тривимірному просторі. Мультипатчі охоплюють плоскі 3D окружності і трикутники, використовувані в комбінації для моделювання тривимірної оболонки. Мультипатчі можуть використовуватися для подання всього, починаючи від простих об'єктів, наприклад сфер і кубів, до складних об'єктів, наприклад з-поверхонь будівель (рис. 2.5).

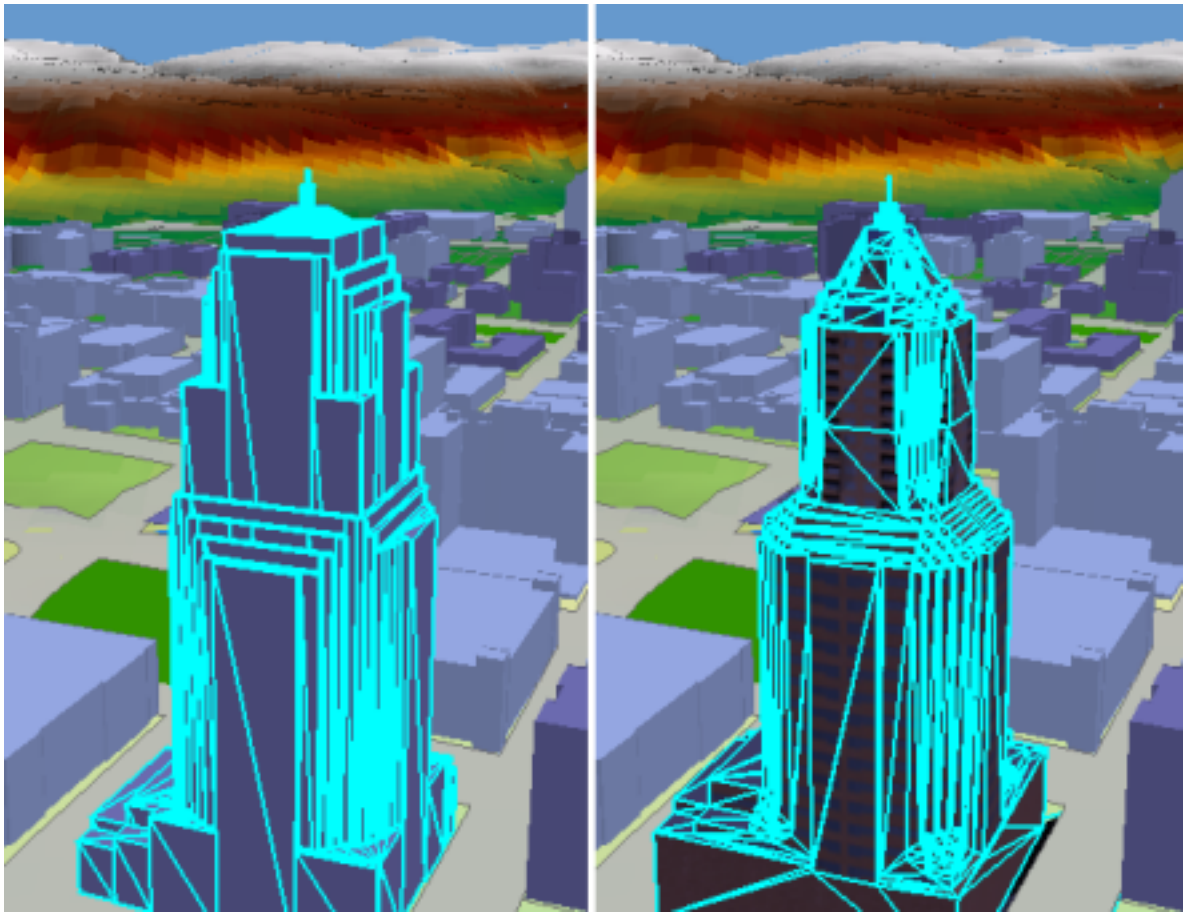

Рисунок 2.5 Приклад просторового об’єкту – Мультипатчу

2.4 Висновок

Найголовнішими функціями Геоінформаційної системи є збір, зберігання, інтеграція, аналіз і графічна інтерпретація просторово-часових даних, а також пов'язаної з ними атрибутивної інформації про представлених в ГІС об'єктах, що робить її невід’ємним помічником в наступних сферах:

- Міське і регіональне планування
- Надзвичайні ситуації та ліквідація стихійних лих
- Дороги і автомагістралі, транспорт
- Економічний розвиток

3. Засоби розробки

3.1 Обґрунтування вибору технологій та їх короткий опис

Реалізація інформаційної системи була здійснена за допомогою мови програмування Swift, де серверна частина написана за допомогою фреймворку ArcGIS Runtime, графічний інтерфейс користувача на нативній бібліотеці UIKit.

Інформаційна система представлена у вигляді мобільного додатку для операційної системи iOS. База даних розміщена у вигляді featureservice (Сервіс об’єктів).

Сервіси об'єктів дозволяють відображати об'єкти через Інтернет і надають символи, що використовуються при відображенні об'єктів. Клієнти можуть виконувати запити для отримання об'єктів і виконувати операції редагування, дозволені на сервері. Сервіси об'єктів надають шаблони, які можна використовувати для розширеного редагування на стороні клієнта. За допомогою сервісу об'єктів також можна виконувати запити в класах відносин і непросторових таблицях і редагувати ці класи і таблиці. [1]

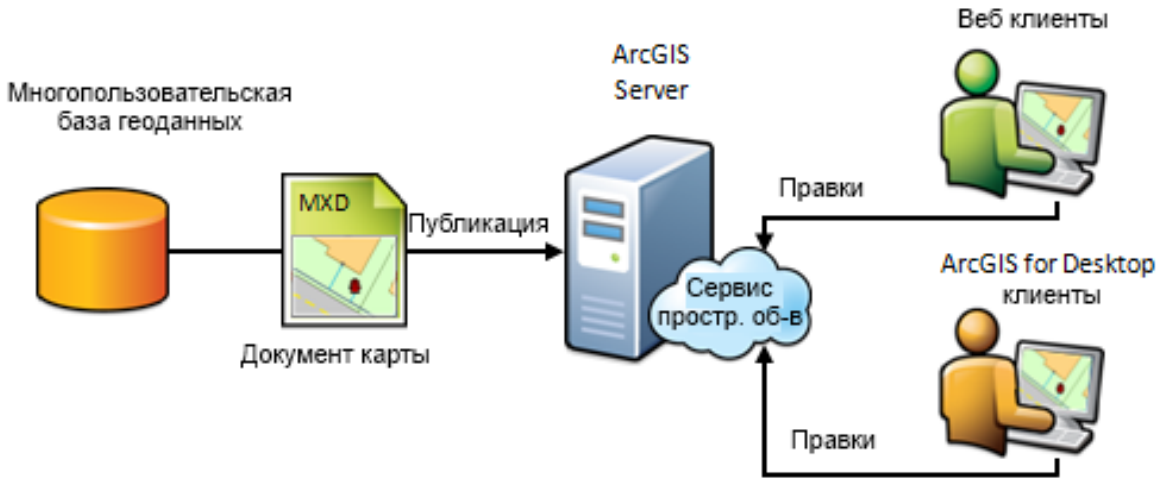

Рисунок 3.1. Робочий процес що використовується для створення сервісу об'єктів

3.2 Опис мови програмування Swift

Swift було створено компанією Apple в першу чергу для розробників iOS та OS X. Вперше дана технологія офіційно вийшла в світ влітку 2014 року. Swift сумісна з основною кодовою базою Apple, написаною на Objective-C. Вона створена таким чином, що може використовуватись разом з Objective-C (а також C) в рамках однієї програми [2].

Синтаксис Swift доволі звичний, мова містить набір інструментів для захисту від помилок та багів. Swift краще розуміє дії автора коду і робить його працю більш ефективною, дозволяє виконати більше завдань за короткий проміжок часу.

Дуже цікавим аспектом даної мови вважаються так звані Ігрові майданчики.

Ігрові майданчики роблять програмування більш інтерактивним і доступним, цікавим та веселим. Зміст полягає в тому, що коли програміст пише код в лівій половині екрану — в правій половині він одразу бачить результат. Таким чином, можна в реальному часі спостерігати за створенням програми. Відповідно, миттєво показуються всі внесені в код зміни без необхідності перезапуску [3].

Swift виключає великий пласт поширених програмних помилок за допомогою застосування сучасних програмних паттернів:

- Змінні завжди ініціалізовані до того, як будуть використані.
- Індекси масивів завжди перевіряються на out-of-bounds помилки.
- Цілі числа перевіряються на переповнення.
- Опціонали гарантують, що значення nil будуть явно оброблені.
- Автоматичне управління пам'яттю
- Обробка помилок дозволяє здійснювати контрольоване відновлення від непередбачених помилок.

Optionals (опціонали) - це зручний механізм обробки ситуацій, коли значення змінної може бути відсутнім. Значення буде використано, лише якщо воно є [4].

3.3 Середовище розробки Xcode

Xcode є інтегрованим середовищем розробки (IDE) для MacOS, набір інструментів для розробки додатків, розроблені Apple, для розробки програмного забезпечення для MacOS, прошивки, iPadOS, watchOS і tvOS. Xcode включає в себе інструменти командного рядка (CLT), які дають можливість розробки під UNIX.

Xcode підтримує вихідний код для мов програмування C, C ++, Objective-C, Objective-C ++, Java, AppleScript, Python, Рубін, ResEdit (Rez) і Swift, з різними моделей програмування, в тому числі, але не обмежуючись ними какао, Вуглець і Java. Треті сторони додана підтримка GNU Pascal, Free Pascal, Ada, C #, Perl, і D.

Xcode може створювати мультиархітектурні бінарні файли, що містять код для декількох архітектур з Mach-O форматом. Вони називаються універсальні виконавчі файли, які дозволяють запускати програмне забезпечення на обох PowerPC і Intel -На (x86) платформах і які можуть включати в себе як 32 і 64-бітний код для обох архітектур. Використання IOS SDK, Xcode також може бути використано для компіляції і налагодження додатків для прошивки, які працюють на ARM архітектурі процесорів.

3.4 Система керування версіями Git

Git - це набір консольних утиліт, які відстежують і фіксують зміни в файлах (найчастіше мова йде про програми вихідного коду). Версія вашого проекту, порівнювати, аналізувати, зливати зміни і багато іншого. Цей процес називається контролем версій.

Git є розподіленим, тобто не залежить від одного центрального сервера, на якому зберігаються файли. Замість цього він працює повністю локально, зберігаючи дані в папках на жорсткому диску, які називаються репозиторієм. Тим не менш, ви можете зберігати копію сховища онлайн, це сильно полегшує роботу над одним

проектом для кількох людей. Для цього використовуються сайти на кшталт github і bitbucket.

Використання Git є необхідною частиною розробки програмного забезпечення, особливо при розробці в команді. Для початку роботи з Git спочатку потрібно завантажити консольну утиліту. Для подальшої роботи з системою контролю версій необхідно зробити наступні кроки:

- За допомогою команди git config вказати ім’я, а також email, щоб інші користувачі могли відстежувати зміни до проекту
- Створити новий репозиторій, перейшовши у відповідну папку з проектом та виконавши команду git init
- Додати файли окремі файли проекту командою git add something.txt, або ж всі файли командою git add -A
- Після внесення змін до файлів проекту необхідно зафіксувати зміни командою git commit -m "Опис змін проекту."

Після Попередніх кроків проект є повністю готовий до розробки, використовуючи системи керування версій Git

4 Опис програмної реалізації

4.1 Опис архітектури

Система побудована на базі продуктів та бібліотеки сімейства ArcGIS. Для початку роботи з системою користувачеві необхідно інсталювати мобільний додаток, та встановити на свій сервер Сервіс об’єктів для геоінформаційної бази даних, або ж завантажити його на хмарний сервіс ArcGIS online, як зробив це я.

Було прийнято рішення розробити систему у вигляді мобільного додатку. Цьому є декілька причин:

- Мобільні пристрої є портативні та інженер дуже просто буде редагувати мережу безпосередньо на місці
- Відсутня потреба у встановлені додаткового програмного забезпечення
- Можливе точне визначення знаходження пристрою

При внесенні змін в базу даних створюється транзакція, що зберігається на сервері протягом деякого часу та може бути скасована адміністратором..

Транзакція — це сукупність операцій, виконаних у логічній послідовності. Кожна транзакція виконується в цілому. Транзакції залишаються невидимі для інших користувачів бази даних, доки до результатів виконання SQL запитів, що входять у транзакцію, не буде застосовано запит до бази даних за допомогою команди COMMIT. Якщо будь-яка частина транзакції не виконується, вся транзакція буде позначена як невдала і не додана в базу даних. Якщо транзакція виконується, то означає, що всі оператори в цій транзакції правильні[6].

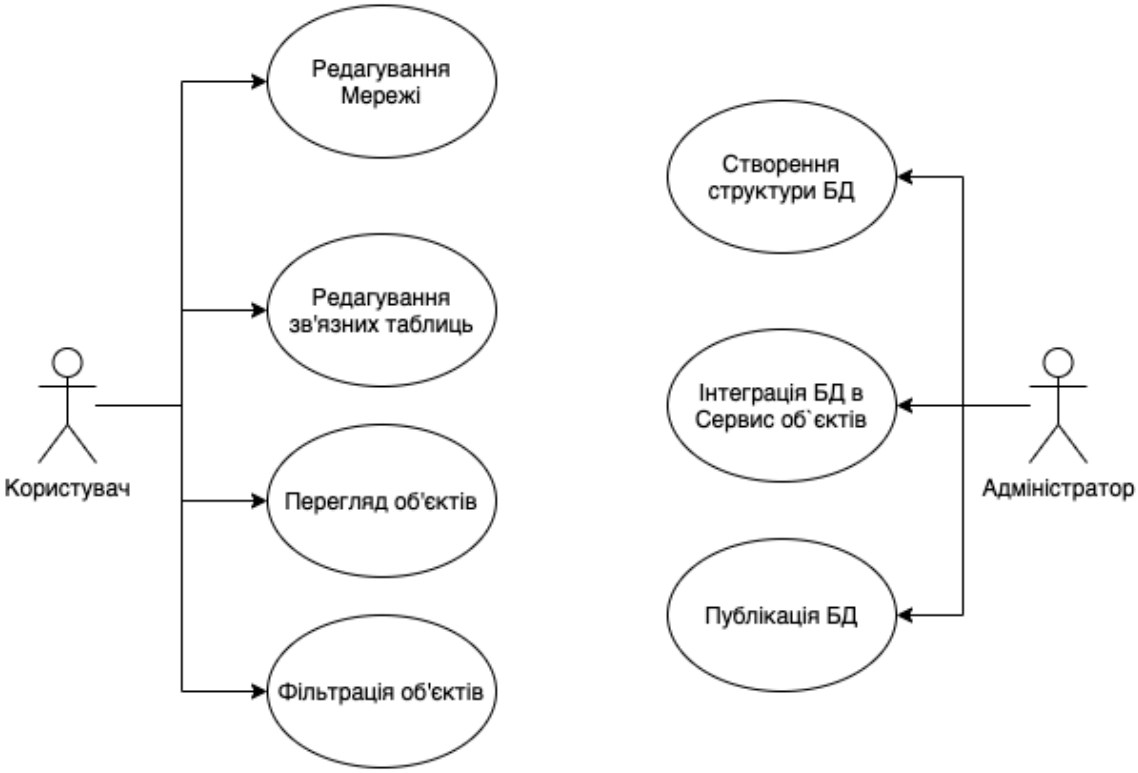

Рисунок 4.1 —Діаграма прецедентів системи

4.2 Створення мобільного додатку

Першочерговою задачею дипломної роботи є створення застосунку для формування геоінформаційної бази даних. Для виконання даного завдання було обрана платформа iOS. Задачею роботи додатку є відображення, редагування та фільтрація об’єктів інженерних мереж.

Створений мобільний застосунок дозволяє користувачеві переглядати, аналізувати та змінювати параметри інженерної мережі, а також додавати нові елементи. При натиску на об’єкти на карті, додаток робить запит на сервер та отримує інформацію про найближчий об’єкт.

Мобільний додаток розроблений на архітектурі MVC (рис 4.2).

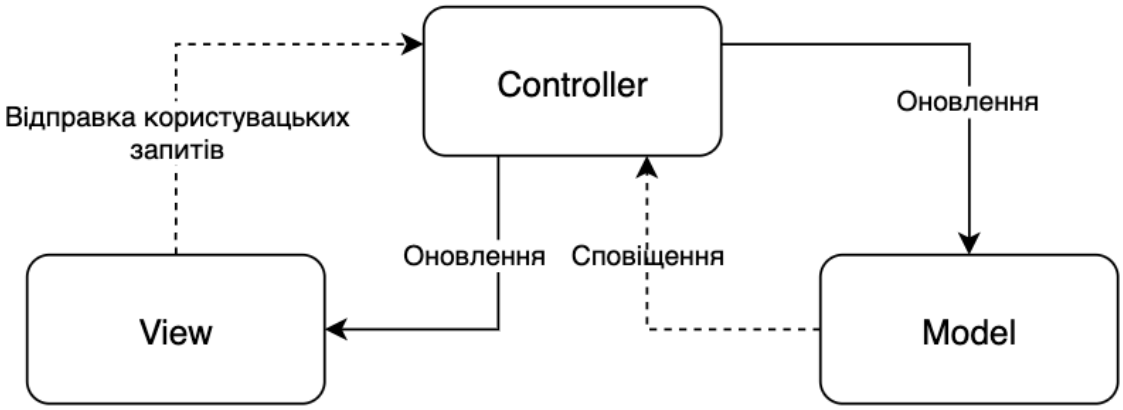

Рисунок 4.2 — Структура архітектури MVC

Суть архітектури полягає у поділу проекту на три модулі:

- Model - відповідальна за дані домену або шар доступу до даних, який маніпулює даними
- View - відповідальні за рівень представлення (GUI); для навколишнього середовища iOS це все, що починається з префікса UI. Каркас графічного інтерфейсу будується в Interface Builder у вигляді ієрархії екранів (рис. 4.3) та зберігається у файлі з розширенням .storyboard
- Controller - посередник між Model і View; в цілому відповідає за зміни Model, реагуючи на дії користувача, виконані на View, і оновлює View, використовуючи зміни з Model.

Маючи розділені суті, можливо краще їх зрозуміти, повторно використовувати, а також тестувати їх окремо один від одного. Controller є посередником між View і Model, отже, дві останніх не знають про існування один одного. Тому Controller важко повторно використовувати. Є також і багато інших складніших архітектурних патернів, таких як MVVM, VIPER та MVP, але для даного проекту був обраний MVC тому, що він є кращим архітектурним патерном з точки зору швидкості розробки.

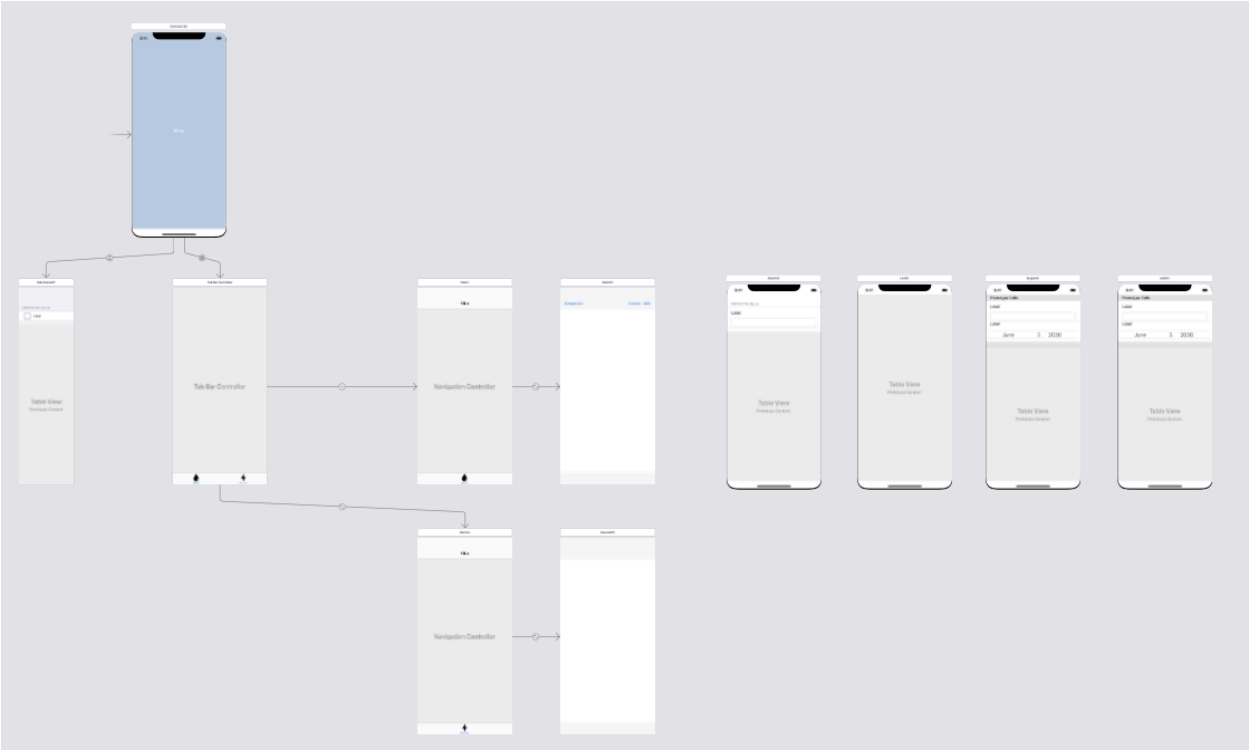

Рисунок 4.3 — Ієрархія екранів інтерфейсу

4.3 Хмарний сервіс для збереження даних ArcGIS online

ArcGISOnline надає місце для розміщення в мережі ваших карт з відповідною графічної інформації, а також обміну нею з користувачами. Це система управління географічною інформацією, що дозволяє обмінюватися вмістом, а також розміщувати його в ГІС додатках і на веб-сайтах кінцевих користувачів. Користувачі підключаються до даних карт і додатків для використання розміщеного вмісту, що керується в хмарі ArcGISOnline, а також в локальних системах.

Інформаційні продукти ArcGISOnline можуть використовуватися для безлічі типів додатків - інтернет, мобільних, додатків Windows і т. Д. ArcGISOnline також дозволяє користувачам, які не є фахівцями в ГІС, створити власні карти і продукти геоданих шляхом змішування шарів, додавання нових шарів з електронних таблиць або простим малюванням на самій карті [8].

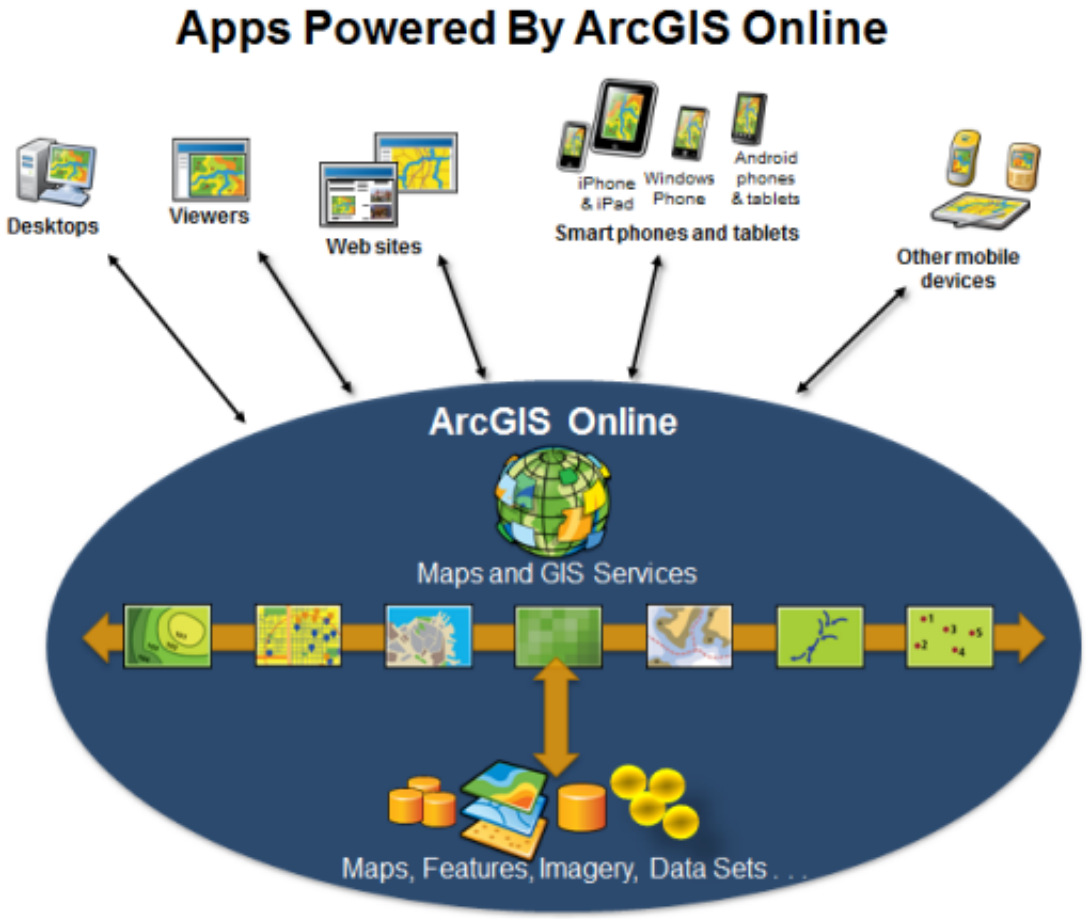

Рисунок 4.4 —Структура ArcGISonline

4.4 Бібліотека ArcGIS Runtime for IOS

ArcGIS Runtime - це технологія розробки настільних і мобільних додатків на платформі ArcGIS. В основі ArcGISRuntime лежить ядро, написане на C ++. До цього ядра є інтерфейси у вигляді API для різних платформ.

Відмінною особливістю додатків ArcGISRuntime є можливість використання цієї функції як в онлайн-режимі за наявності підключення до мережі (за рахунок веб-сервісів ArcGISOnline або корпоративного ГІС-сервера ArcGISfor Server), так і в офлайн-режимі за відсутності мережевого підключення (за рахунок локальних бібліотек). При цьому з точки зору програмування робота з віддаленими веб-сервісами та локальними сервісами не відрізняється. При переході з онлайн-режиму в офлайн-режим і назад підтримується консистентність даних.

5 Робота користувача з програмною системою

5.1 Системні вимоги

Для роботи з додатком накладаються вимоги з продуктивності роботи апаратного забезпечення.

Для роботи мінімально необхідні: Мобільний пристрій на базі операційної системи iOS 12 і вище.

5.2 Робота розробника з системою

Для тестового запуску системи необхідно мати комп’ютер на базі операційної сисеми macOS та останню версію середовища розробки Xcode. Система одразу налаштована для роботи з інженерними мережами НТУУ «КПІ ім. Ігоря Сікорського», але розробник може налаштувати її під свої потреби виконавши наступну послідовність дій:

- Зареєструвавшись на порталі arcGIS необхідно завантажити попередньо архівовану в zip форматі ГБД натиснвши кнопу “Add Item” в пункті «Content» (рис. 5.1). Після цього відповідна ГБД завантажиться та створиться сервіс об'єктів.

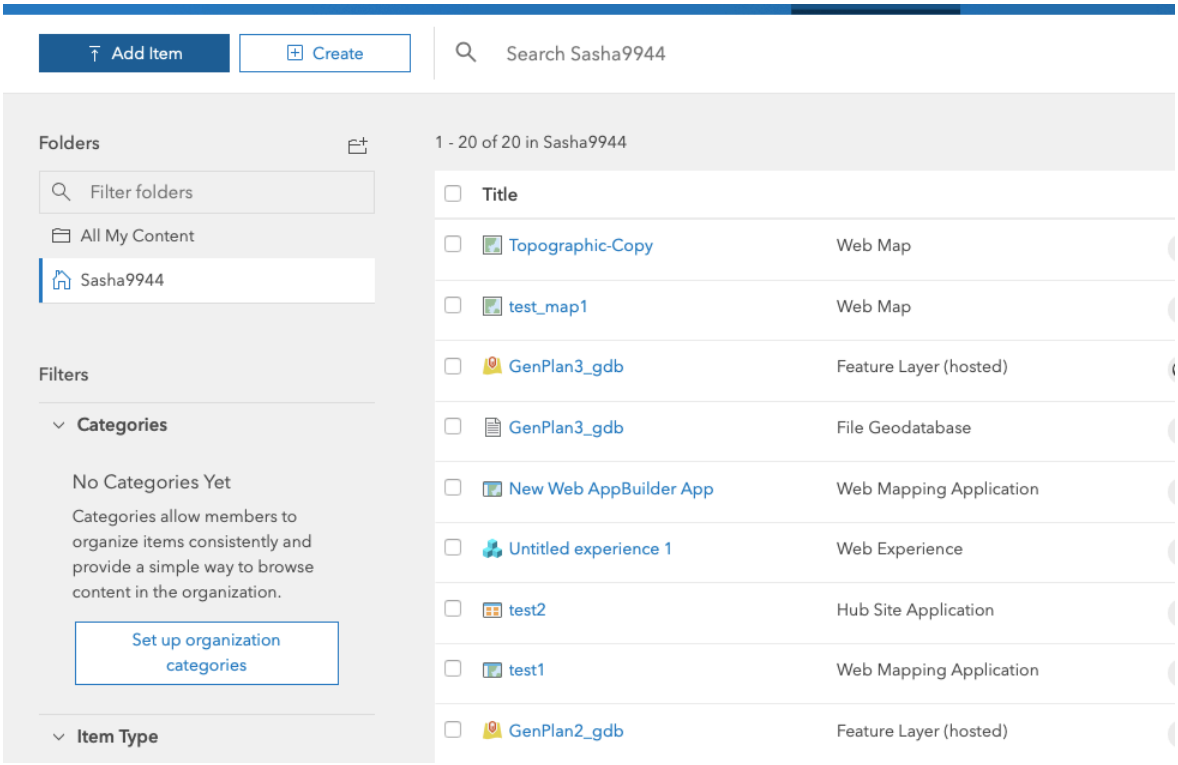

Рисунок 5.1 — Екран порталу ArcGIS

- Наступним кроком потрібно відкрити файл AppConfig.swift в проекті та замінити url сервіса обєктів на щойно завантажений в класі MapConfig (рис. 5.2)

```
class MapConfig {

    let featureServiceString =
        "https://services7.arcgis.com/6LV5dTlFXsOot0n8/arcgis/rest/services/GenPlan3_gdb/FeatureServer"
    var featureServiceURL: URL {
        return URL(string: featureServiceString)!
    }

    func getFeatureTable(by id: TableID) -> AGSServiceFeatureTable {
        guard let featureServiceURL = URL(string: featureServiceString + "/" + String(id.rawValue)) else {
            fatalError("Error: Problem with FeatureServer in AppConfig")}
        return AGSServiceFeatureTable(url: featureServiceURL)
    }

}
```

Рисунок 5.2 — Код класу MapConfig

- Після цього необхідно обрати які просторові об’єкти та зв’язні таблиці користувач хоче бачити на мапі вказавши їх id у відповідному перерахуванні (рис. 5.3)

```
enum TableID: Int {
    case waterMeter = 18
    case waterPipes = 3
    case waterHatch = 8
    case waterCounter = 22
    case waterMeasure = 23
    case electroMeter = 20
    case electroPipes = 17
    case electroHatch = 11
    case electroCounter = 24
    case electroMeasure = 21
    case electroCompare = 25
    case buildings = 1
}
```

Рисунок 5.3 — Код перерахування TableID

- Потім використовуючи клас LayerDesigner (рис 5.4) надаємо об’єктам відповідного вигляду (рис 5.5)

```
class LayerDesigner {

    func designLayerRenderMark(layer: AGSFeatureLayer, style: AGSSimpleMarkerSymbolStyle, color: UIColor, size:
        CGFloat) {
        let uniqueValueRenderer = AGSUniqueValueRenderer()
        let symbol = AGSSimpleMarkerSymbol(style: style, color: color, size: size)
        uniqueValueRenderer.defaultSymbol = symbol
        layer.renderer = uniqueValueRenderer
        layer.renderingMode = .dynamic
    }

    func designLayerRenderLine(layer: AGSFeatureLayer, style: AGSSimpleLineSymbolStyle, color: UIColor, width:
        CGFloat) {
        let uniqueValueRenderer = AGSUniqueValueRenderer()
        let symbol = AGSSimpleLineSymbol(style: style, color: color, width: width)
        uniqueValueRenderer.defaultSymbol = symbol
        layer.renderer = uniqueValueRenderer
        layer.renderingMode = .dynamic
    }

}
```

Рисунок 5.4 — Код класу LayerDesigner

```
private func setupMap() {
    let map = AGSMap(basemapType: .navigationVector, latitude: 50.449680, longitude: 30.454492, levelOfDetail:
        17)

    tableHatch = appConfig.getFeatureTable(by: .waterHatch)
    tableMeter = appConfig.getFeatureTable(by: .waterMeter)
    tablePipes = appConfig.getFeatureTable(by: .waterPipes)
    tableCounter = appConfig.getFeatureTable(by: .waterCounter)
    tableMeasure = appConfig.getFeatureTable(by: .waterMeasure)

    layerHatch = AGSFeatureLayer(featureTable: tableHatch)
    layerMeter = AGSFeatureLayer(featureTable: tableMeter)
    layerPipes = AGSFeatureLayer(featureTable: tablePipes)

    layerDesigner.designLayerRenderMark(layer: layerHatch, style: .square, color: .blue, size: 11)
    layerDesigner.designLayerRenderMark(layer: layerMeter, style: .triangle, color: .blue, size: 11)
    layerDesigner.designLayerRenderLine(layer: layerPipes, style: .solid, color: .blue, width: 2)

    map.operationalLayers.add(layerHatch!)
    map.operationalLayers.add(layerPipes!)
    map.operationalLayers.add(layerMeter!)
    map.tables.addObjects(from: [tableCounter!, tableMeasure!])
    mapView.touchDelegate = self
    mapView.callout.delegate = self

    self.mapView.map = map
}
```

Рисунок 5.5 — Функція конфігурації карти

Після попередніх дій залишилось підключити телефон до комп’ютера, зібрати додаток та запустити на мобільному пристрою. Тепер додаток готовий до роботи з новою базою даних.

5.3 Результати виконання програми

Запустивши програму з мобільного пристроюю натиснувши на іконку додатку, спочатку запуститься екран загрузки (рис. 5.6)

Рисунок 5.6 — Екран загрузки

Після повного завантаження додатку відкривається стартовий екран (рис 5.7)

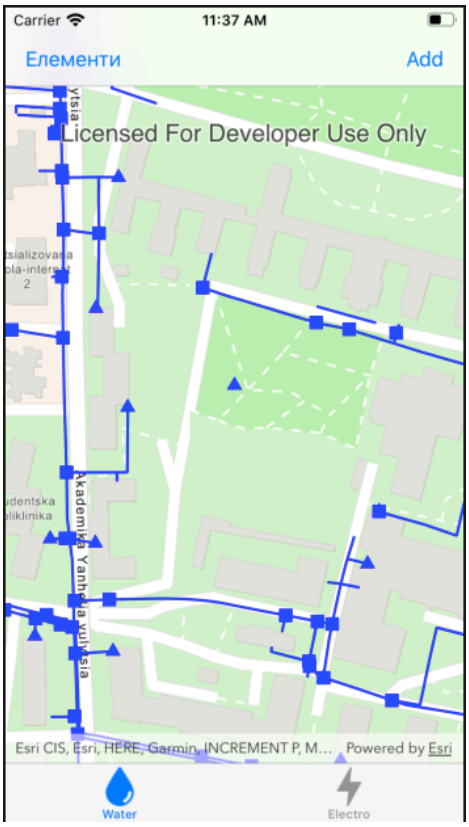

Рисунок 5.7 — Стартовий екран

Для зручного відображення та фільтрації шарів в верхньому кутку є кнопка «Елементи» для відкриття бокового меню (рис 5.8). При натиску на відповідний пункт, шар з’явиться на карті або приховається.

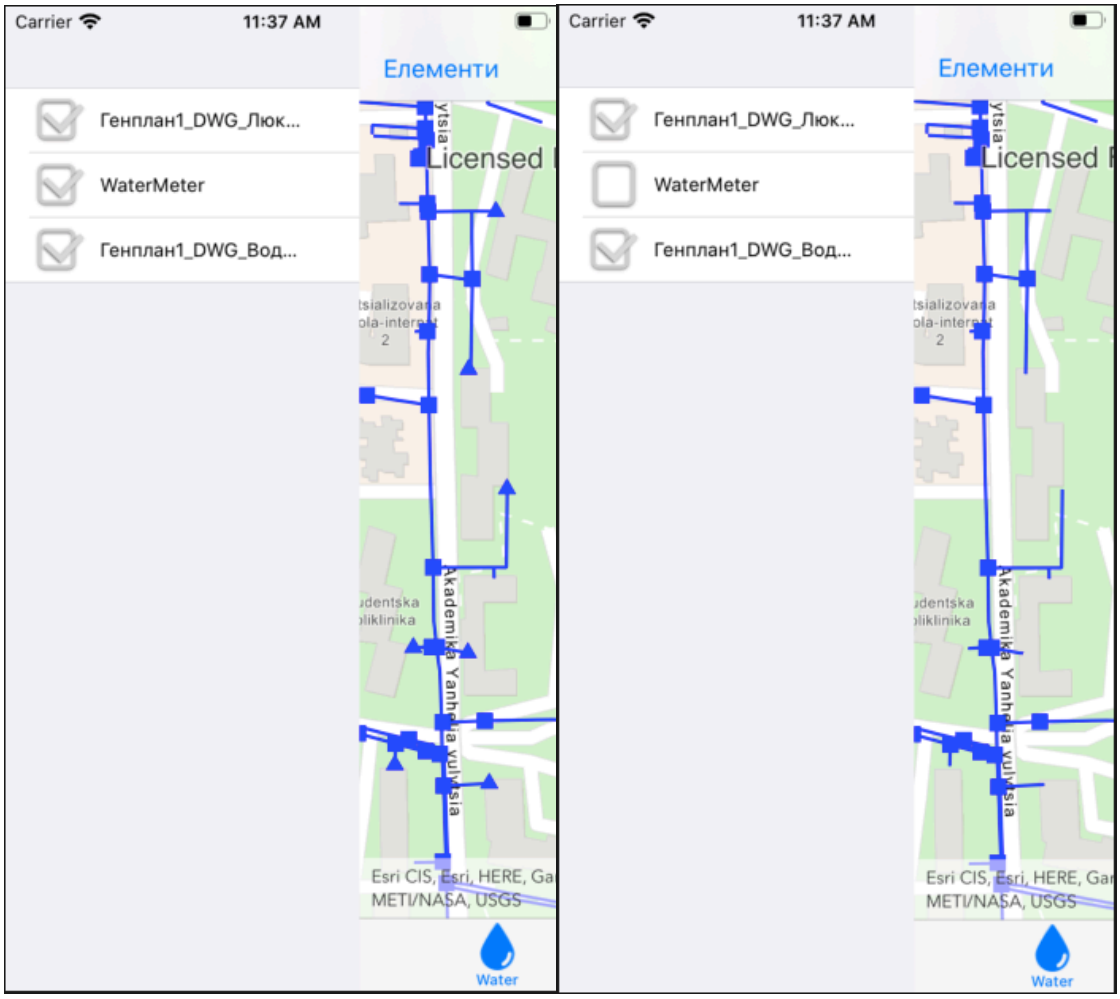

Рисунок 5.8 — Бокове меню

При натиску на певний шар над ним відкривається виноска з кнопкою детальної інформації (рис 5.9)

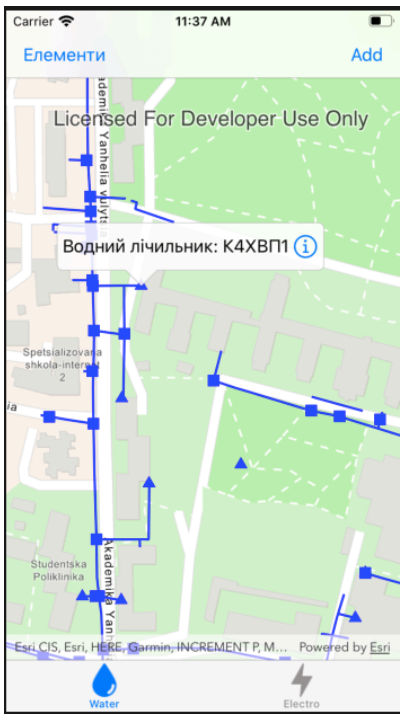

Рисунок 5.9— Відображення виноски водного лічильника

При відкритті детальної інформації користувач може побачити поля відповідного шару а також зв’язні таблиці (рис 5.10).

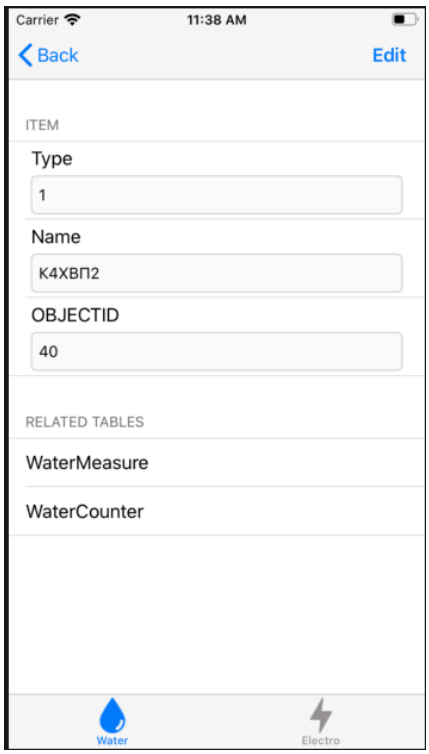

Рисунок 5.10 — Інформація про шар

Користувач має змогу відредагувати інформацію про шар натиснувши на кнопку “Edit”. Після цього поля стануть активними а назва кнопки зміниться на «Done» (рис. 5.11).

Рисунок 5.11 — Активований режим редагування

З екрану детальної інформації про шар можна переглядати зв’язні поля з інших таблиць натиснувши на відповідні пункти в секції «Related Tables».

Якщо зв’язок між шаром та таблицею один до одного то відкриється схожий екран до попереднього, а в даному випадку він буде містити детальну інформацією про лічильник (рис 5.12). Якщо ж зв’язок з таблицею один до багатьох то відкриється екран зі списком зв’язних елементів, а в даному випадку відобразяться список вимірювань певного лічильника (рис 5.13), з подальшою можливістю додавання нових елементів.

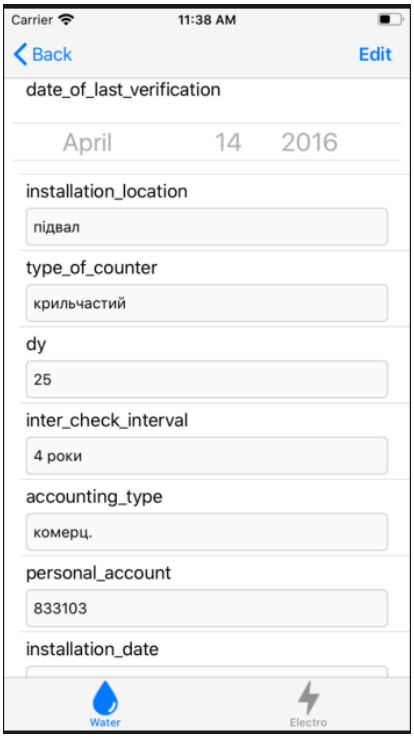

Рисунок 5.12 — Детальна інформація про лічильник

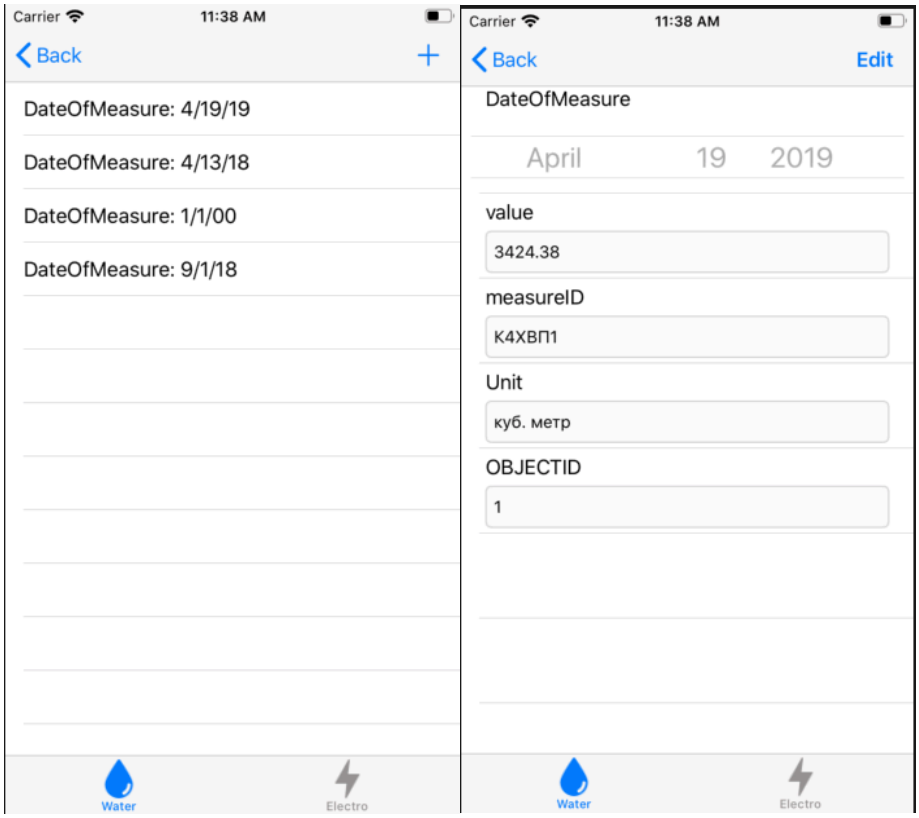

Рисунок 5.13 —Перегляд зв’язної таблиці показників лічильника та інформації про певний лічильник

При натисканні на кнопку «+» користувач може додати новий елемент до спису, а в даному випадку це внесення нового показника в лічильник (рис. 5.15).

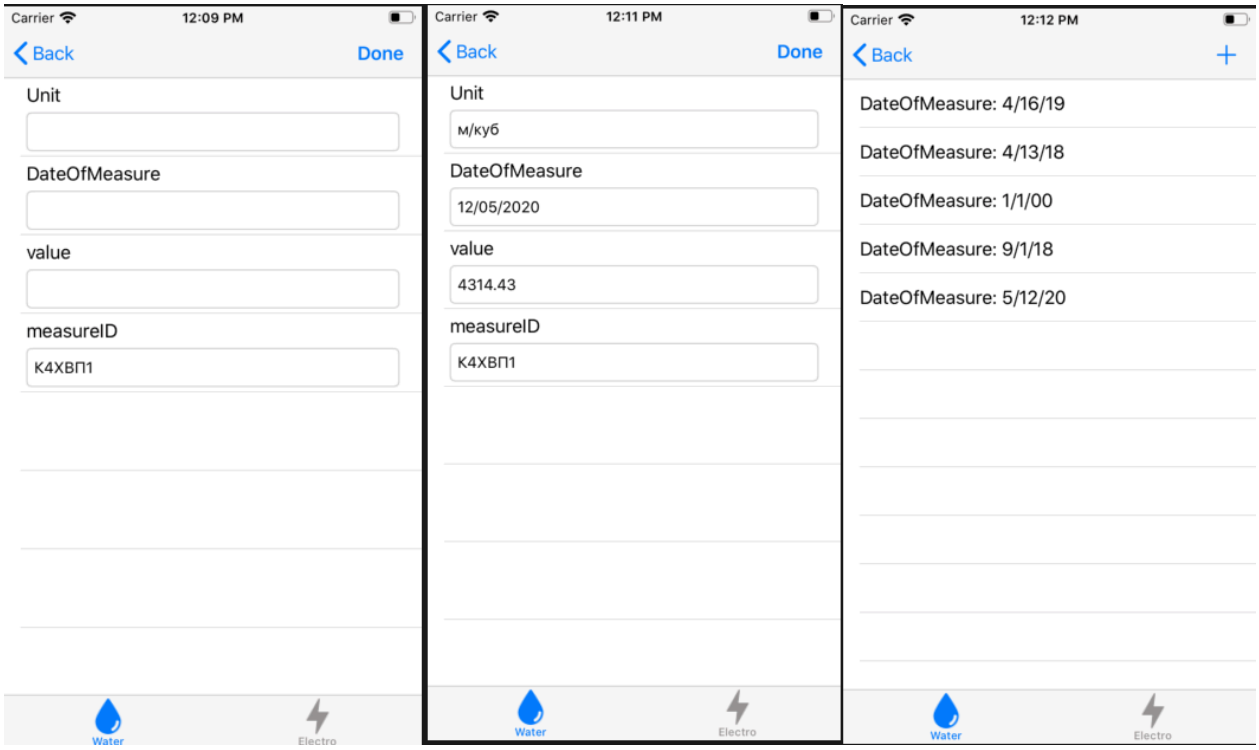

Рисунок 5.14 —Додавання нового показника

Для додавання нового елементу на карту необхідно на стартовому екрані натиснути кнопку “Add”. Після чого відкриється контекстне меню з переліком можливих шарів для додавання (рис. 5.15).

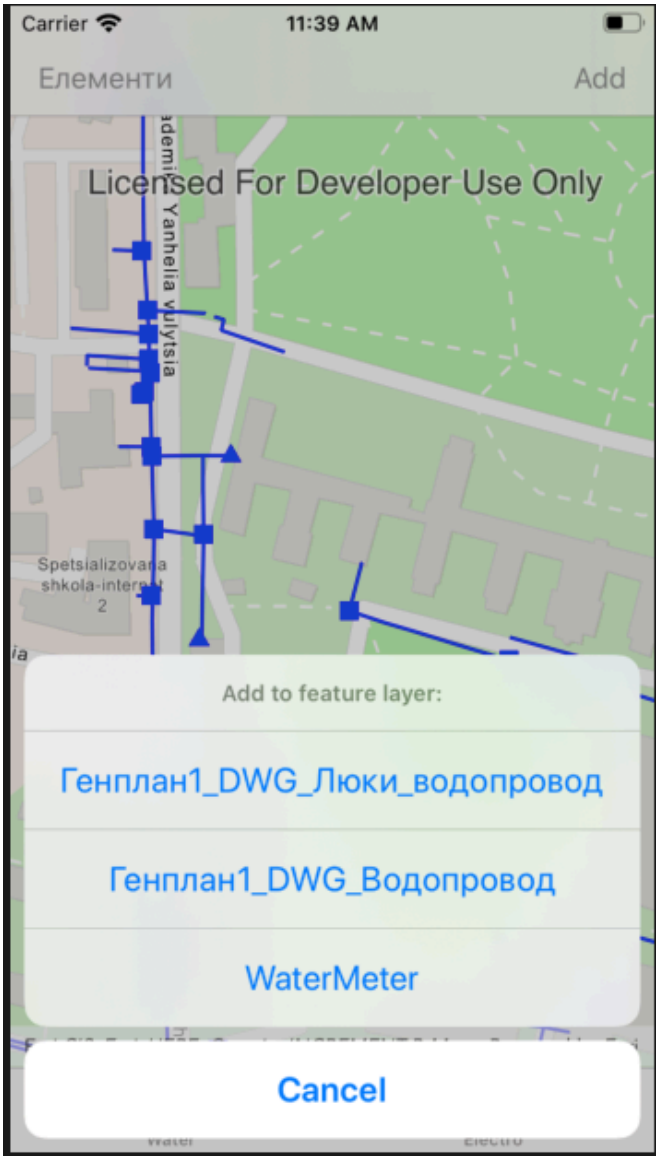

Рисунок 5.15 — Відкрите контекстне меню

Після обрання бажаного шару, в центрі екрану з’явиться мітка розташування майбутнього об’єкту (рис 5.16). Рухаючи карту та масштаб, користувач обирає необхідне місце для встановлення об’єкту. Для підтвердження створення об’єкту необхідно натиснути кнопку «Done» або ж «Cancel» для відміни. Після підтвердження відкриється екран для заповнення атрибутів об’єкту (рис. 5.17). Якщо користувач правильно заповнив поля то елемент успішно збережеться та відобразиться на карті, і потім можна буде заповнити його зв’язні таблиці.

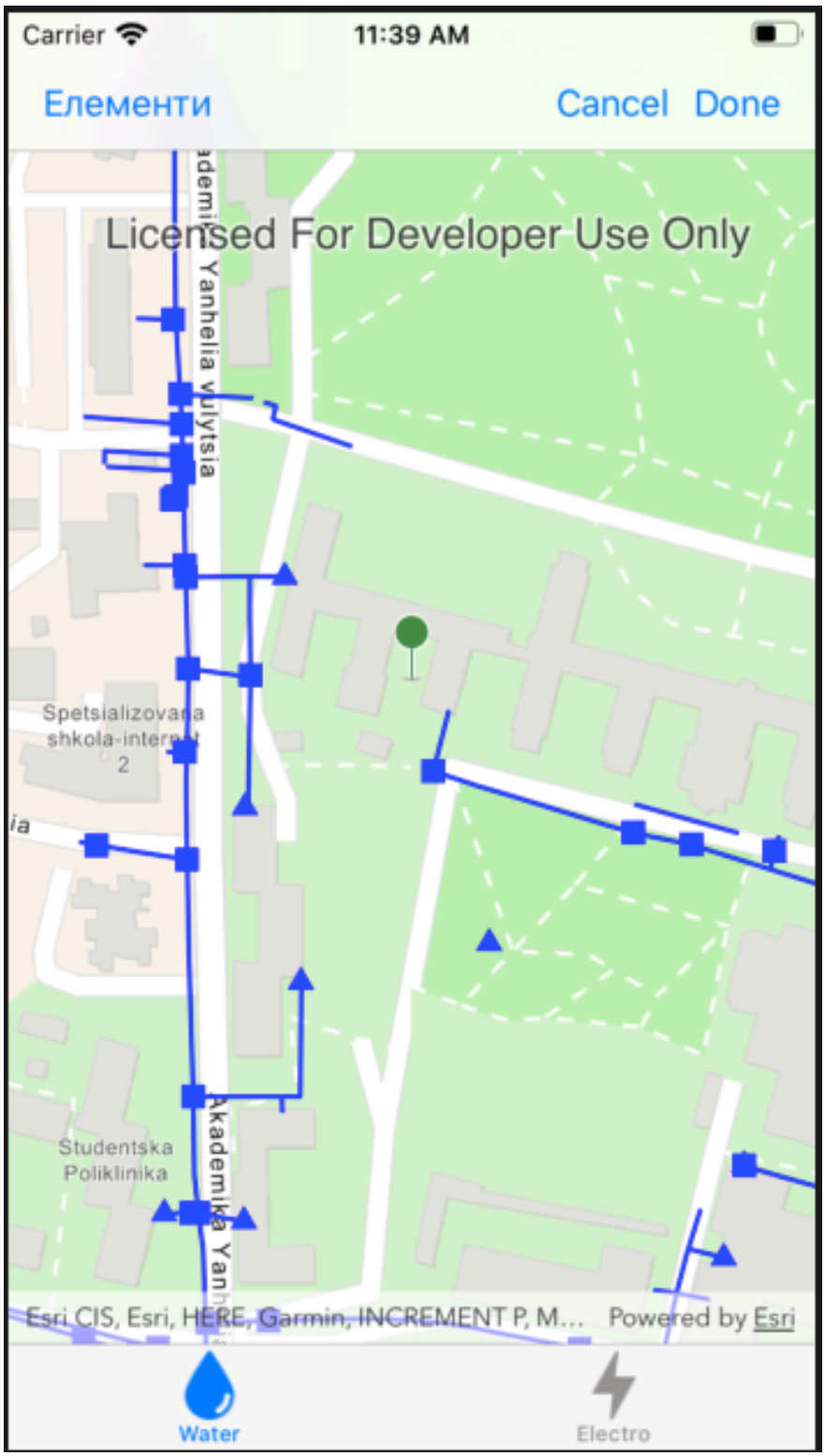

Рисунок 5.16 — Відображення мітки на карті

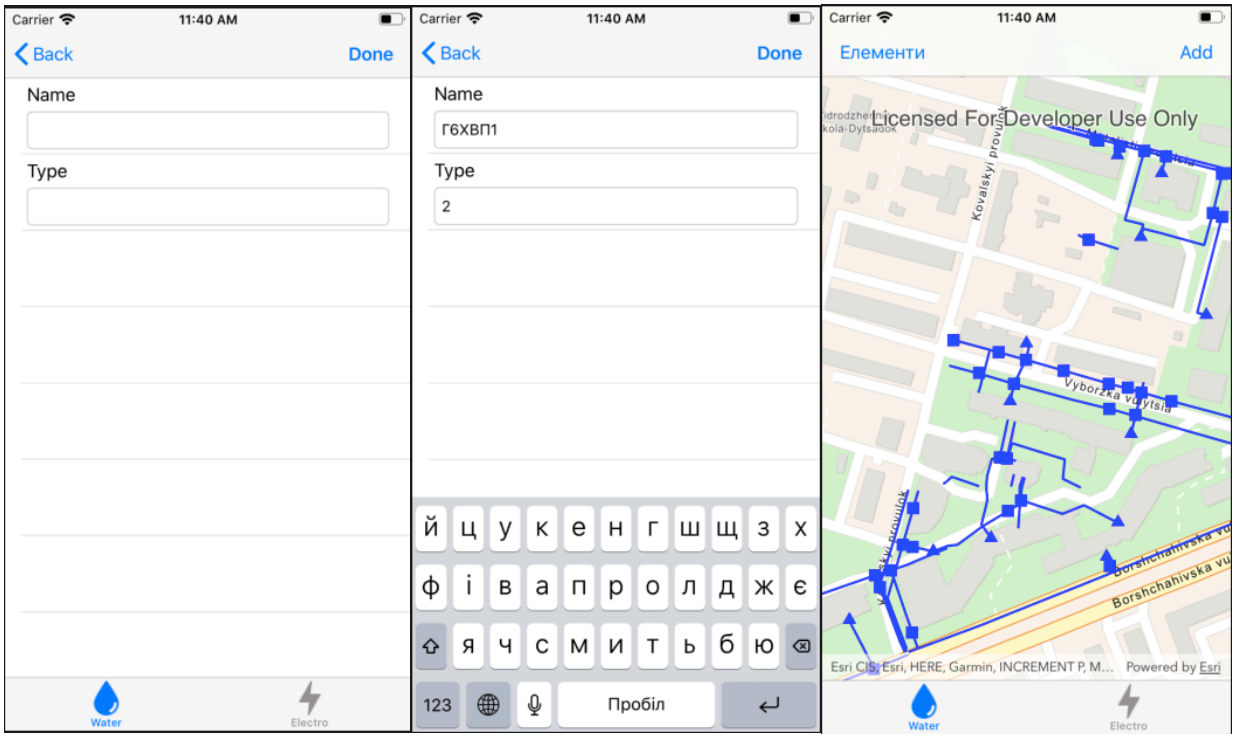

Рисунок 5.17 — Внесення даних про шар та його відображення

Проект легко масштабується, тому аналогічний функціонал реалізований і для електромережі (рис. 5.18)

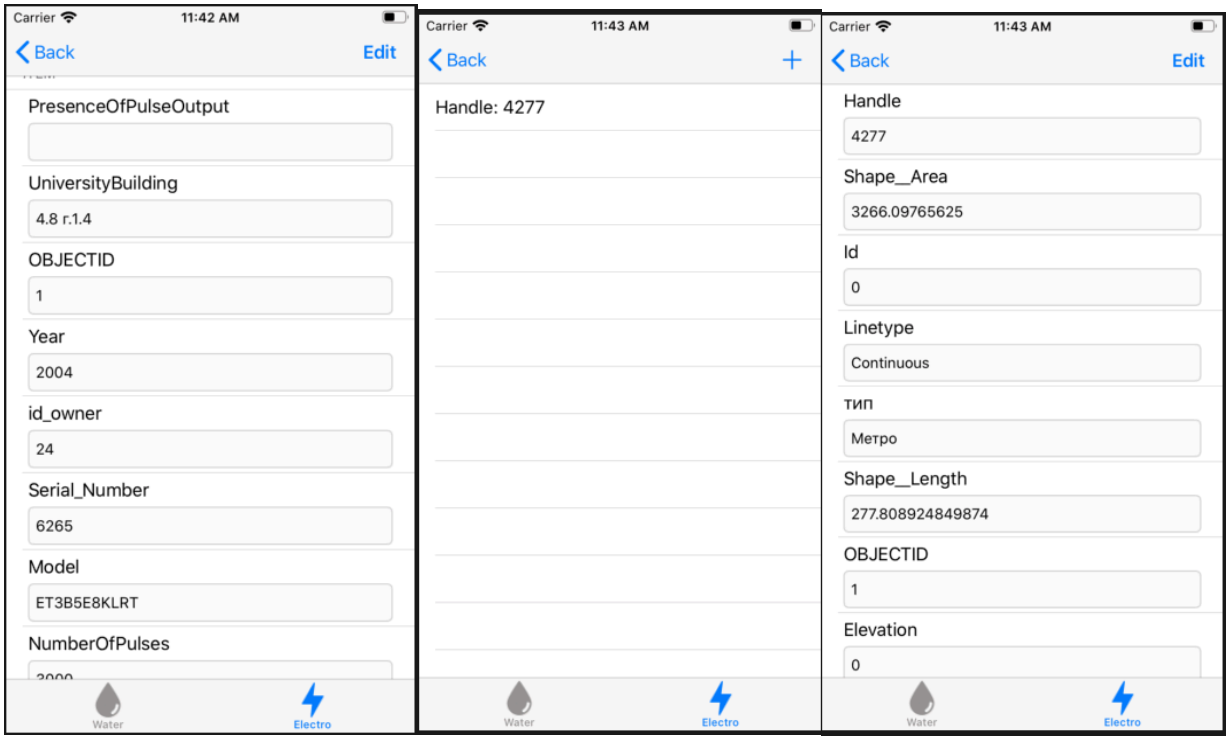

Рисунок 5.18 — Перегляд таблиць електромережі

ВИСНОВОК

Під час виконання роботи був проведений аналіз предметної області. У результаті чого була створена система для редагування та аналізу інженерних мереж. Під час створення системи було проаналізовано та вивчено з чого складаються інженерні мережі. Були покращені навички мобільної розробки та маніпуляції просторових даних. В систему було занесено дані про інженерні мережі студмістечка: водопостачання, енергопостачання. Архітектура системи побудована таким чином, щоб легко розширюватись з потребами користувача. Оскільки геоінформаційна база даних знаходиться на сервері, то в систему можуть бути просто інтегровані додаткові модулі для розширення функціоналу. Це можуть бути наступні модулі:

- Система що збирає дані з лічильників, трансформаторів, датчиків
- Система що аналізує навантаження на енергомережу
- Система що аналізує тиск водопостачання
- Система для створення аналітичних звітів

Практичне значення системи полягає у полегшені роботи енергоменеджера розширення його можливостей використовуючи інформаційні технології. Система допустимо може використовуватись інженерами та менеджерами для автоматизації обліку енергомереж, створення оперативної аналітики та статистики по зібраних даних з мережі.

СПИСОК ДЖЕРЕЛ

1. Что такое сервис объектов? [Електронний ресурс] – Режим доступу до ресурсу: https://enterprise.arcgis.com/ru/server/10.3/publish-services/windows/what-is-a-feature-service-.htm.

2. Swift [Електронний ресурс] – Режим доступу до ресурсу: https://swift.org/about/#swiftorg-and-open-source

3. О языке Swift [Електронний ресурс] – Режим доступу до ресурсу: https://swiftbook.ru/content/swift-tour/about-swift/.

4. Опционалы в Swift [Електронний ресурс] – Режим доступу до ресурсу: https://habr.com/ru/post/338766/.

5. ArcGIS Runtime [Електронний ресурс] – Режим доступу до ресурсу: https://www.esri-cis.ru/products/arcgis-runtime/detail/review/.

6. Транзакції в SQL [Електронний ресурс] – Режим доступу до ресурсу: https://www.quality-assurance-group.com/sql-pytannya-na-spivbesidi-chastyna-3/.

7. Что такое база геоданных? [Електронний ресурс] – Режим доступу до ресурсу: https://desktop.arcgis.com/ru/arcmap/10.3/manage-data/geodatabases/what-is-a-geodatabase.htm.

8. Что такое ArcGIS Online? [Електронний ресурс] – Режим доступу до ресурсу: https://doc.arcgis.com/ru/arcgis-online/get-started/what-is-agol.htm.

9. Классы пространственных объектов.Краткий обзор [Електроннийресурс].– Режим доступу: http://desktop.arcgis.com/ru/arcmap/10.3/manage-data/feature- classes/a-quick-tour-of-feature-classes.htm

10. Міські інженерні мережі: Навч. посібник (для студентів 4, 5, 6 курсів спец. 7.092102 – «Міське будівництво і господарство», 7.120103 – «Містобудування» та напряму 1201 – «Архітектура»). / Деркач І. Л. // Харків: ХНАМГ – 2006.– 97 с.

11. Определение свойств класса пространственных объектов [Електронний ресурс]. – Режим доступу: http://desktop.arcgis.com/ru/arcmap/10.3/manage-data/feature-classes/defining-feature-class-properties.htm

12. ДеМерс М. Географические информационные системы. Основы. / Майкл ДеМерс., 1999. – 478 с.

13. Баранов Ю. Б. Геоинформатика: толковый словарь основных терминов / Ю. Б. Баранов, А. М. Берлянт, Е. Г. Капралов и др. – М. : ГИСАссоциация, 1999. – 204 с.

14. Архитектура базыг еоданных [Електронний ресурс] – Режим доступу до ресурсу: https://desktop.arcgis.com/ru/arcmap/10.3/manage-data/geodatabases/the-architecture-of-a-geodatabase.htm.

15. Праховник А.В., Іншеков Є.М. Визначенні термінів і одиниць виміру та аналіз енергетичної ситуації. Вісник Сумського державного університету. Серія «Економіка». 2006

ДОДАТОК 1

Створення програмних засобів формування бази даних інженерних мереж на базі ГІС технологій

Специфікація

УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ61170_20Б

Аркушів 2

Київ – 2020

Позначення	Найменування	Примітки
Документація		
УКР.НТУУ «КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕ ПС_ТМ61170_20Б	Козачук О. В. ТМ-61.docx	Пояснювальна зписка
Компоненти		
УКР.НТУУ «КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕ ПС_ТМ61170_20Б	Project.mxd /Campus.gdb/	Модулі геоінформаційної бази даних
УКР.НТУУ «КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕ ПС_ТМ61170_20Б	KPI manager/ KPI manager.xcworkspace KPI manager/ KPI manager/ ViewControllers/ MainVC.swift KPI manager/ KPI manager/ ViewControllers/ SingleVC.swift	Модулі мобільного додатку

ДОДАТОК 2

Створення програмних засобів формування бази даних інженерних мереж на базі ГІС технологій

Текст програми

УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ61170_20Б

Аркушів 10

Київ – 2020

```
//
//  MainVC.swift
//  KPI manager
//
// MARK: Клас головного екрану
class MainVC: UIViewController {

    // MARK: Допоміжні сервіси
    private let graphicsOverlay = AGSGraphicsOverlay()
    let appConfig = MapConfig()
    let layerDesigner = LayerDesigner()

    // MARK: Змінні шарів
    var layerHatch: AGSFeatureLayer!
    Var layerMeter: AGSFeatureLayer!
    Var layerPipes: AGSFeatureLayer!

    // MARK: Змінні таблиць
    var tableHatch: AGSServiceFeatureTable!
    Var tableMeter: AGSServiceFeatureTable!
    Var tablePipes: AGSServiceFeatureTable!
    Var tableCounter: AGSServiceFeatureTable!
    Var tableMeasure: AGSServiceFeatureTable!

    Var selectedTable: AGSServiceFeatureTable?
    Var stoage = FeatureStoarge.shared
    var isAdding: Bool = false {
        didSet {
            if isAdding {
                cancelButton.title = “Cancel”
                cancelButton.isEnabled = true
                rightButton.title = “Done”
            } else {
                cancelButton.title = “”
                cancelButton.isEnabled = false
                rightButton.title = “Add”
            }
        }
    }
    var addingFeature: AGSFeatureLayer?

    // MARK: Змінні інтерфейсу
    @IBOutlet weak var mapView: AGSMapView!
    @IBOutlet weak var pinDropView: PinDropView!
    @IBOutlet weak var rightButton: UIBarButtonItem!
    @IBOutlet weak var cancelButton: UIBarButtonItem!

    Override func viewDidAppear(_ animated: Bool) {
        super.viewWillAppear(animated)
        updateLayers()
    }

    // MARK: Функція що викликається при відкритті екрану
    override func viewDidLoad() {
        super.viewDidLoad()
        mapView.graphicsOverlays.add(graphicsOverlay)
        setupMap()
        setupPin()
```

```
        cancelButton.title = “”
        cancelButton.isEnabled = false
    }

    // MARK: Функція конфігурації мітки
    func setupPin() {
        pinDropView.pinDropped = false
    }

    // MARK: Функція кнопки cancel
    @IBAction func cancel(_ barButtonItem: UIBarButtonItem?) {
        isAdding = false
        pinDropView.pinDropped = false
    }

    // MARK: Функція що викликається при запиті користувача на додавання нового шару
    @IBAction func userRequestsAddNewFeature(_ barButtonItem: UIBarButtonItem?) {
        if isAdding  {
            let centerPoint = mapView.centerAGSPoint
            guard let featureLayer = self.addingFeature else { return }
            guard let table = featureLayer.featureTable as? AGSServiceFeatureTable
else { return }
            isAdding.toggle()
            let vc = AddVC.instantiate(from: .main)
            vc.configure(table: table, point: centerPoint)
            navigationController?.pushViewController(vc, animated: true)
            pinDropView.pinDropped = false
        } else {
            guard let map = mapView.map, let layers = (map.operationalLayers as?
[AGSFeatureLayer]), layers.count > 0 else {
                present(simpleAlertMessage: “No eligible feature layer that you can
add to.”)
                return
             }

            if layers.count == 1  {
                // There is only one eligible layer, begin the process of adding a
new feature for that layer.
                addNewFeatureFor(featureLayer: layers.first!)
                return
             }
             else {
                // Find the layer to which the new record will be added.
                Let action = UIAlertController(title: nil, message: “Add to feature
layer:”, preferredStyle: .actionSheet)

                for layer in layers {

                    guard let featureTable = layer.featureTable as? AGSArcGISFea-
tureTable else {
                        continue
                    }

                    let addFeature = UIAlertAction(title: featureTable.tableName,
style: .`default`, handler: { [weak self] (action) in
                        self?.addNewFeatureFor(featureLayer: layer)
                    })

                    action.addAction(addFeature)
                }
```

```
                action.addAction(.cancel())

                action.popoverPresentationController?.barButtonItem = barButtonItem
                present(action, animated: true, completion: nil)
            }
        }
    }

    // MARK: Функція що сповіщає додаток про створення новго шару
    private func addNewFeatureFor(featureLayer: AGSFeatureLayer) {
        pinDropView.pinDropped.toggle()
        isAdding.toggle()
        self.addingFeature = featureLayer
    }

    // MARK: Функція оновлення шарів
    func updateLayers() {
        stoage.featureLayer = []
        stoage.featureLayer.append(contentsOf: [layerHatch, layerMeter, layerPipes])
    }

    // MARK: Функція конфігурації мапи
    private func setupMap() {
        let map = AGSMap(basemapType: .navigationVector, latitude: 50.449680, longi-
tude: 30.454492, levelOfDetail: 17)

        tableHatch = appConfig.getFeatureTable(by: .waterHatch)
        tableMeter = appConfig.getFeatureTable(by: .waterMeter)
        tablePipes = appConfig.getFeatureTable(by: .waterPipes)
        tableCounter = appConfig.getFeatureTable(by: .waterCounter)
        tableMeasure = appConfig.getFeatureTable(by: .waterMeasure)

        layerHatch = AGSFeatureLayer(featureTable: tableHatch)
        layerMeter = AGSFeatureLayer(featureTable: tableMeter)
        layerPipes = AGSFeatureLayer(featureTable: tablePipes)

        layerDesigner.designLayerRenderMark(layer: layerHatch, style: .square,
color: .blue, size: 11)
        layerDesigner.designLayerRenderMark(layer: layerMeter, style: .triangle,
color: .blue, size: 11)
        layerDesigner.designLayerRenderLine(layer: layerPipes, style: .solid, color:
.blue, width: 2)

        map.operationalLayers.add(layerHatch!)
        map.operationalLayers.add(layerPipes!)
        map.operationalLayers.add(layerMeter!)
        map.tables.addObjects(from: [tableCounter!, tableMeasure!])
        mapView.touchDelegate = self
        mapView.callout.delegate = self

        self.mapView.map = map
    }

    // MARK: Функція що викликає бокове меню
    @IBAction func menuButtonTapped() {
        NotificationCenter.default.post(name: NSNotification.Name(“ToggleSideMenu”),
object: nil)
    }
```

```
}

// MARK: Розширення делегат для збору та аналізу натисків на мапу
extension MainVC: AGSGeoViewTouchDelegate {

    // MARK: Функція що аналізує натиски на екран
    func geoView(_ geoView: AGSGeoView, didTapAtScreenPoint screenPoint: CGPoint,
mapPoint: AGSPoint) {

        self.mapView.identifyLayer(self.layerMeter, screenPoint: screenPoint, toler-
ance: 12, returnPopupsOnly: false, maximumResults: 10) { [weak self] (result: AG-
SIdentifyLayerResult) -> Void in

            if let error = result.error {
                print(error)
                return
            }

            if let feature = result.geoElements.first as? AGSArcGISFeature {
                self?.select(feature, atLocation: mapPoint)
            } else {
                self?.deselect()
            }
        }

        self.mapView.identifyLayer(self.layerHatch, screenPoint: screenPoint, toler-
ance: 12, returnPopupsOnly: false, maximumResults: 10) { [weak self] (result: AG-
SIdentifyLayerResult) -> Void in

            if let error = result.error {
                print(error)
                return
            }

            if let feature = result.geoElements.first as? AGSArcGISFeature {
                self?.select(feature, atLocation: mapPoint)
            } else {
                self?.deselect()
            }
        }
    }


    // MARK: Функція для виділення лічильника
    private func select(_ feature:  AGSArcGISFeature, atLocation location: AGSPoint)
{
        let title = feature.attributes[“Name”] as? String
        mapView.callout.title = “Водний лічильник: “ + (title ?? “undefinded”)
        mapView.callout.isAccessoryButtonHidden = false
        mapView.callout.show(for: feature, tapLocation: location, animated: true)
    }

    // MARK: Функція для виділення люка
    private func selectHatch(_ feature:  AGSArcGISFeature, atLocation location: AG-
SPoint) {
        let title = feature.attributes[“Handle”] as? String
        mapView.callout.title = “Люк: “ + (title ?? “undefinded”)
        mapView.callout.isAccessoryButtonHidden = false
        mapView.callout.show(for: feature, tapLocation: location, animated: true)
```

```
    }

    // MARK: Функція відміни виділення
    private func deselect() {
        mapView.callout.dismiss()
    }


}


// MAPK: Розширення делегат для виклику аннотації
extension MainVC: AGSCalloutDelegate {
    func didTapAccessoryButton(for callout: AGSCallout) {
        if let feature = callout.representedObject as?  AGSArcGISFeature, let table
= feature.featureTable as? AGSServiceFeatureTable {
            self.selectedTable = table
            let vc = DetailVC.instantiate(from: .main)
            //vc.configure(feature: feature, relatedFeatures: selectedFeatures)
            //vc.configure1(feature: selectedFeatures)
            vc.configure2(feature: feature, table: table)
            navigationController?.pushViewController(vc, animated: true)
            self.deselect()
        }
    }
}
//  SingleVC.swift
//  KPI manager
// MAPK: Клас для відображення та редагування обраного об’єкту
class SingleVC: UIViewController {

   @IBOutlet weak var tableView: UITableView!
   Var dataSource: [TableColumn] = []
   var feature: AGSArcGISFeature!
   Var table: AGSServiceFeatureTable?

   Var isEditable: Bool = false {
      didSet {
         tableView.reloadData()
      }
   }

   override func viewDidLoad() {
      super.viewDidLoad()
      setupMenu()
   }

   // MAPK: Функція конфігурації меню
   func setupMenu() {
      let editButton = UIBarButtonItem(title: “Edit”, style: .done, target: self, action: #selector(editBut-
tonTapped))
      navigationItem.rightBarButtonItem = editButton
   }

   // MAPK: Функція що викликається при зміні статусу редагування
```

```
    @objc func editButtonTapped() {
        if isEditable {
            let cells = tableView.getAllCells(rowCount: dataSource.count, section: 0)

            for (index, cell) in cells.enumerated() {
                let item = dataSource[index]
                let key = item.title
                if item.type == .date {
                    guard let dateCell = cell as? DateCell else { return }
                    let valuePicker = dateCell.datePicker.date
                    guard let value = item.getDate() else { return }

                    if value != valuePicker {
                        feature.attributes[key] = valuePicker
                        dataSource[index].value = item.formatter.string(from: valuePicker)
                        tableView.reloadRows(at: [IndexPath(row: index, section: 0)], with: .automatic)
                    }
                } else {
                    guard let textFieldCell = cell as? TextFieldCell else { return }
                    guard let testFieldText = textFieldCell.textField.text else { return }
                    guard let value = item.getValue(string: testFieldText) else { return }
                    let stringValue = item.value

                    if testFieldText != stringValue {
                        dataSource[index].value = testFieldText
                        feature.attributes[key] = value
                    }
                }

            }

            self.table?.update(feature, completion: { (error:Error?) -> Void in

                if let error = error {
                    print("Error while updating feature :: \(error.localizedDescription)")
                    return
                }
                self.table?.applyEdits(completion: { (featureEditResults: [AGSFeatureEditResult]?, error: Er-
ror?)-> Void in
                    if let error = error {
                        print("Error while applying edit :: \(error.localizedDescription)")
                        return
                    }
                    print("Apply Edits successful")
                    self.tableView.reloadData()
                })
            })


        }
```

```
    isEditable.toggle()
    navigationItem.rightBarButtonItem?.title = isEditable ? “Done” : “Edit”
}

// MARK: Функція що викликається при закритті вікна
override func viewWillDisappear(_ animated: Bool) {
    super.viewWillDisappear(animated)
    table?.clearCache(withKeepLocalEdits: false)
}

// MARK: Функція конфігурації екрану через об’єкт
func configure(feature: AGSArcGISFeature) {
    self.feature = feature
    if let table = feature.featureTable as? AGSServiceFeatureTable {
        self.table = table
    }

    let data = feature.attributes

    for (key, value) in data {
        self.dataSource.append(TableColumn(key: key, value: value))
    }


}

// MARK: Функція конфігурації екрану через результат вибірки
func configure(result: AGSRelatedFeatureQueryResult, idTable: Any) {

    if result.featureEnumerator().allObjects.count > 0 {
        guard let relatedTable = result.relatedTable as? AGSServiceFeatureTable else { return }
        guard var url = relatedTable.url else { return }
        let urlSeseion = URLSession.shared
        var urlComponents = URLComponents(string: url.absoluteString)!

        urlComponents.queryItems = [
            URLQueryItem(name: “f”, value: “json”)
        ]

        if let url1 = urlComponents.url {
            url = url1
        }

        let task = urlSeseion.dataTask(with: url) { [weak self] data, response, error in
            do {
                let relTable: Table = try JSONDecoder().decode(Table.self, from: data!)
                guard let relationship = relTable.relationships.first else { return }
                let idRelatedTable = relationship.keyField
```

```
                let queryParameters = AGSQueryParameters()
                queryParameters.whereClause = "\(idRelatedTable) like '%\(idTable)%'"
                //queryParameters.returnGeometry = true
                let outFields: AGSQueryFeatureFields = .loadAll
                relatedTable.queryFeatures(with: queryParameters, queryFeatureFields: outFields) { [weak self] result, error in

                    if let error = error {
                        print("Error querying the trailheads feature layer: \(error.localizedDescription)")
                        return
                    }

                    guard let result = result, let features = result.featureEnumerator().allObjects as? [AGSArcGISFeature] else {
                        print("Something went wrong casting the results.")
                        return
                    }



                    for item in features {
                        self?.feature = item
                        let data = item.attributes

                        for (key, value) in data {
                            self?.dataSource.append(TableColumn(key: key, value: value))
                        }
                    }

                    DispatchQueue.main.async {
                        self?.tableView.reloadData()

                        if let table = features.first?.featureTable as? AGSServiceFeatureTable {
                            self?.table = table
                        }
                    }
                }

            } catch {
                print("JSON error: \(error.localizedDescription)")
            }
        }

        task.resume()
    }
  }
}

// MARK: Розширення DataSource для забеспечення даними таблиці
```

```
extension SingleVC: UITableViewDataSource {

    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
        dataSource.count
    }

    func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell
{
        let item = dataSource[indexPath.row]
        if item.type == .date {
            let cell = tableView.dequeueReusableCell(withIdentifier: “DateCell”, for: indexPath) as! DateCell
            cell.setup(tableColumn: item, isEnabled: isEditable)
            return cell
        } else {
            let cell = tableView.dequeueReusableCell(withIdentifier: “TextFieldCell”, for: indexPath) as!
TextFieldCell
            cell.setup(tableColumn: item, isEnabled: isEditable)
            return cell
        }

    }


}
```

ДОДАТОК 3

Створення програмних засобів формування бази даних інженерних мереж на базі ГІС технологій

Опис програми

УКР.НТУУ«КПІ ім. Ігоря Сікорського»_ТЕФ_АПЕПС_ТМ61170_20Б

Аркушів 8

Київ – 2020

АНОТАЦІЯ

Додаток надає можливість редагувати параметри шарів та зв’язних таблиць шару, а також додавати нові елементи на шар.

Програмний продукт розроблений на мові програмування Swift для платформи iOS з використанням фреймворку ArcGIS Runtime. Розробка інтерфесу додатку відбувалась за допомогою фреймворку UIKit та Interface Builder. В якості середовища розробки було використано – XCode.

ЗМІСТ

ЗАГАЛЬНІ ВІДОМОСТІ

Програма має назву Kpi Manager тому, що для тестової бази даних був створений прототип бази даних інженерної мережі студмістечка КПІ.

Програма працює з мобільного пристрою з операційною системою iOS і потребує підключення до мережі інтернет для доступу до бази даних.

Система була написана мовою Swift.

ФУНКЦІОНАЛЬНЕ ПРИЗНАЧЕННЯ

Розроблений програмний засіб покликаний вирішити задачу формування бази даних інженерних мереж. Це було реалізовано за допомогою кількох функцій системи, а саме:

- Перегляд шарів інженерної мережі
- Редагування зв'яних таблиць об'єктів інженерної мережі
- Додавання нових точкових об'єктів на шар інженерної мережі

ОПИС ЛОГІЧНОЇ СТРУКТУРИ

Загальний принцип роботи додатку такий:

1) Користувач переглядає попередньо завантажену на сервер базу даних
2) Користувач натискає на карті на певний об’єкт
3) Після натиску, відображається коротка інформація про об’єкт
4) Після натиску на кнопку додаткової інформації користувач відкриє повну інформацію про об’єкт і його зв’язні таблиці, та також змінити або добавити певні елементи
5) З головного екрану користувач може також обрати шар, потім місце для встановлення нового об’єкту

Програма містить два головних екрани що змінюються між собою при натиску на елементи навігаційного меню знизу екрану. В них міститься інформація про Інженерні мережі водо- та електропостачання у вигляді окремих мап. Також в додатку існують ще 3 основні додаткові екрани що перевикористовуються для попередніх двох. Вони служать для відображення та редагування інформації про одиночні об’єкти, одиночні об’єкти зі зв’язними таблицями а також списки об’єктів.

ВИКОРИСТОВУВАНІ ТЕХНІЧНІ ЗАСОБИ

Для організації доступу до програмного продукту потрібно мати комп'ютер на базі операційної системи Mac OS а а також телефон з операційною системою iOS.

Кінцевим користувачам, окрім самого мобільного додатку, не потрібно встановлювати додаткового програмного забеспечення.

ВХІДНІ І ВИХІДНІ ДАНІ

Вхідними даними є:

- Геоінформаційна база даних завнатажена на сервер у вигляді сервісу об’єктів
- Натиск пальцем або стилусом на карту
- Маркери відображення шару
- Інформація з текстових полів

Вихідними даними є:

- Інформація про об'єкти на шарах
- Інформація про об'єкти в зв'язних таблицях
- Графічне відображення інженерних мереж